Structure, curvature and movement

Nsugbe, Emma Ada Obiageli

Award date:
2001

Awarding institution:
University of Bath

Structure, Curvature and Movement

The Application of Organicism to Architecture using mathematics and Computer Programming to generate form

Submitted by Emma Ada Obiageli Nsugbe

For the degree of Doctor of Philosophy of the University of Bath

2001

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

'The most expert Artists among the Ancients... were of [the] Opinion that an Edifice was like an Animal, so that in the Formation of it we ought to imitate Nature.'

L.B.Alberti, Ten Books on Architecture, trans. J.Leoni (London, 1955), book 9, p194.

Acknowledgements

This work would not have been possible without the continual love and support of my family, in particular the encouragement and patience of my parents.

I am indebted to my supervisor, Dr.C.J.K.Williams for his patience and persistence in explaining mathematical concepts that were not so familiar to me as an architect. I am grateful also to Professor Mike Barnes for his interest and support especially at the transfer stage.

I would like to express appreciation to Sir Michael Hopkins & Partners, Snell Associates and Feilden Clegg Architects who provided part-time work, which helped to fund the earlier part of my period of study.

And finally, financial support is gratefully acknowledged from the University of Bath, The Esther Parkin Trust and the British Federation of Women Graduates for awards made to me in the latter stages of this work making its completion possible.

For Mum and Dad

Synopsis

This research considers the question of generating architectural and engineering form using mathematics and computer programming as the investigative method.

The impetus for the research arises from a philosophy about design, which is guided by nature as ultimate designer and producer of shape.

The case for design in nature is first of all presented in chapter 1, then a palette of nature's tools and architectural precedent follow in chapter 2. The mathematical and computer interfacing is then examined in chapter 3, which prepares the ground for an exploration of the methods in chapter 4. Chapter 4 selects and presents 2 design studies, which result in an expression of architecture and engineering form that seeks to capture the essence of natural geometry. A third study, dealing with key aspects of the form generation of The British Museum roof, is discussed in the appendix.

In conclusion, it is argued in chapter 5 that the methods are valid and worthwhile ways to create architectural and engineering form, and that although at first might present barriers to an unfamiliar readership, promise new opportunities for making complex curvilinear shapes in a computer advancing world.

Table of Contents

List of Illustrations

CHAPTER 1.0

CHAPTER 2.0

Introduction

The question of ascribing mathematical logic or giving artistic expression to the geometry and process of the natural world is one that has been explored by philosophers, mathematicians, artists, scientists and writers from the time of Pythagoras. It is a universal quest, which sets out to extract an understanding of form and law in the natural world that could provide a model for creativity. Whyte (1955) calls this law of nature *'the Unknown Formative Principle, the Logical Structure of Becoming, the Logos Chronos.'* I call it the elusive force and shape of life.

This quest to find laws to account for form and order in the universe has led to significant advancements in physics and mathematics, which have contributed to developments in geometry. These developments have in turn enabled the definition of complex forms in design.

This research continues the quest and investigates its potential for architecture. It aims to demonstrate the use of mathematical methods and computer programming as a method of generating form in architecture using nature as source of reference.

In architecture and engineering, concepts that draw inspiration from nature follow the organic tradition of William Morris, Louis Sullivan and Antoni Gaudí. Architectural theorists such as Caroline van Eck (1994) have adopted the term *organicism* to describe the use of nature as metaphor in art and architecture.

In the twentieth century, a philosophy of organicism is evident in the work of architects such as Le Corbusier, Frank Lloyd Wright, Buckminster Fuller, Frei Otto, Santiago Calatrava and Frank Gehry, who have all used nature as the source for an aesthetic. Texts relating to the work and writings of these architects and theorists whose ideology is underpinned by organicism have been reviewed in formulating a theoretical framework for this research.

The physical attributes of natural forms provide a very broad scope, perhaps too broad a scope for detailed study, and their characteristics are clearly innumerable. This research focuses on their aesthetic and dynamic qualities, namely of structure, curvature and movement.

Implicit in curved forms are notions of dynamism and change. Curved forms are composed of elements, which are continually changing direction thus appearing to be in motion. Pettigrew (1908), Cook (1904) and others have observed that animals in motion, either walking, swimming or flying have a tendency to plot curvilinear paths. The forms of plants and animals, their transformation and transposition through growth and movement are intensely governed by curvilinear characteristics. Curvature is therefore associated with life form and governs its essential life function. It is firmly embedded in the double helix of our DNA.

Historically, mathematics has been associated with methods for describing curved and fractal geometry in nature, such as the logarithmic spiral of mollusc shells and the branching structures of wing membranes and trees. Mathematical ratios have been used to represent and predict the harmony, consistency and proportion that plants, animals and physical matter show in growth and movement. The Fibonacci series, for example, suggests a template for spiral phyllotaxis, a feature found in the leaf arrangements of plants.

The introduction of computer technology has led to important implications for architectural design, both in the method of drawing production and in the method of producing building components. Manual drawing and model-making techniques have thus far dominated the method of capturing form. They will indeed remain, at least in the initial stages of the design process, as a means to express a concept graphically. However, the increasing complexity of three-dimensional geometry calls for new ways to represent form. This representation of complex geometry lies beyond the scope of manual techniques if both accurate and detailed shape information and structural analysis are required. Flynn (1999) has observed that a lack of mathematical knowledge has hampered the discovery and practice of a more fluid architecture. However through the increasing availability of software incorporating mathematical tools, dynamic morphology is becoming more accessible to the architect.

Gehry, whose Guggenheim Museum in Bilbao has been referred to as the metallic flower, approaches the problem of realising complex curvilinear forms by making physical models. The physical models are refined by further sculpting, until finally, they achieve the

desired result. The physical models are then accurately digitised and the digitised data forms the basis for computer drawings, which are then further manipulated using computer-modelling packages. These techniques however, remain essentially manual and the computer is used as a form enhancing tool.

This research aims to investigate various methods of defining curved geometry in relation to the generation of form in architecture. In order to investigate this problem, three design processes which took place at the University of Bath are studied. The first study focuses on defining the curvilinear geometry of branching bone-like forms. This is applied to the design of two bridges, a wall form and a sculpture.

Branching requires the transition from a cross-section of one closed curve to two or more separate closed curves, which is a property of the maps produced by complex analytic functions in two dimensions. Therefore complex analysis is used in this particular study as the main tool in the geometric definition of branching structures. Maps derived from complex functions produce curvilinear quadrilaterals, which have the advantage of bringing orthogonal geometry into curvilinear systems and vice versa. Because the geometry is derived mathematically, manipulations and distortions can be achieved by overlaying one function onto another. Unlike the second and third studies, the results of the first study have not been built.

The second study is the Rest Zone in the Millennium Dome by the Richard Rogers Partnership and Buro Happold. This research examines a method using a combination of mathematics and computer programming to define the geometry required by the architects, which in this case was a modified torus form.

The third study, which is included in Appendix D, describes the techniques used to generate the geometry of the glass roof structure over the British Museum Great Court by Foster and Partners and Buro Happold. In this study, a method is devised to create a grid-shell of triangular components, which translate the circular geometry of the Sydney Smirke Reading Room near the centre of the Court, into the rectilinear geometry of the perimeter walls. Spirals are used to reconcile the apparent conflict between circular and rectilinear geometry.

The approach adopted in this thesis is both historical and technological. It is historical in the sense that it examines the historical and philosophical context under which the ideology of organicism is formulated, and it is technological in the sense that it tests the use of this ideology in design practice.

The objective of the work is to encourage greater accessibility and dissemination of the methods to a wider audience of architects, engineers and designers. The nature of the investigation is inter-disciplinary and a collaborative approach to design is heavily relied upon; an approach that is reflected in current trends in the architectural and engineering professions which benefit considerably from previous inter-disciplinary scholarship and now, from advanced computer technology.

Chapter 1.0 The History of Organicism in Architecture

The aim of this chapter is to present the historical context in which organicist world-views developed. It begins by describing organicism as a system of thought, which dictated the direction and character of ancient philosophical enquiry. Enquiry at this time had been underpinned by Aristotelian values, which were later abandoned for more scientific methods of enquiry, by pioneering minds such as those of Leonardo da Vinci and Galileo.

The eventual deepening of scientific knowledge, and the infiltration of scientific forms of organicism into other areas of creative endeavour, had a direct impact on the development of architectural style in the late nineteenth century. Various traditions of organicism have persisted into the twentieth and twenty-first centuries, in continually re-moulded forms.

The latter part of this chapter recounts the lineage and diversity of the expression of organicism in architecture, from the nineteenth century practitioners of the *Art Nouveau*, through to the modernism of Le Corbusier, the sculptural traditions of Candela and Nervi, and on to the structural traditions of Buckminster Fuller.

1.1 Background

1.1.1 The Meaning of Organicism

The term organicism does not appear in the Concise Oxford Dictionary 1990. Its components, *organic* and *ism*, however, do. Thus organicism refers to a *'system, principle, ideology, doctrine or practice'* associated with *'organic structures, organisms or organised physical structures'* and can be applied to a variety of disciplines.

Caroline van Eck (1994) in her book, entitled *Organicism in nineteenth-century architecture*, uses the word organicism to describe a theory of organic form and she discusses the application of this theory within the context of nineteenth century architecture. In the introduction to her book, she refers to organicism as one of the most widespread and constant themes in the history of Western architecture and its theory, but also one of the most variable and elusive. Any attempts to define its absolute meaning are thwarted with ambiguity.

Notwithstanding the pitfalls associated with defining organicism, van Eck nevertheless does attempt a definition of it describing it as *'the metaphorical application to architecture of concepts originally reserved for living nature.'* According to her, organicism encourages an accord between artistic and scientific invention and the methods and forms of the natural world. It is a philosophy that ascribes to nature the task of guiding the endeavours of art and science.

The term organicism and van Eck's definition of it has been adopted in this thesis because it makes clearer the notion of organicism as a philosophy that could govern style rather than the term organic architecture, which has connotations of being a style in itself. Bornstein (1996) in *The Structurist*, agrees that organic creation in art or architecture should not be referred to as a style, but rather as the product of a principle - a biological, morphological and structural principle which is concerned with the processes of formation rather than the sylistic product, and can therefore take a variety of shapes and directions. It has pervaded a range of periods in architectural history from the Romanesque to the Gothic Revival and the Art Nouveau.

Organicism is open to wide interpretation and should not be confined within the limits of a style. It is an ideological instrument that transcends style.

1.1.2 Organicism and Ancient Enquiry

The notion of organicism dates from antiquity. It was embedded in a philosophy of nature that existed at the time of Aristotle when artistic and scientific thought was dominated by the search for knowledge about the causes of natural phenomena.

Organicism involved a concern to establish and maintain a close connection between the natural world (the classic model of an organic whole), and our existence within it. It was an all-encompassing creed, a universal bud reared from the seed of Aristotelian belief. It shaped the expression of art and steered the direction of science whilst seeking to explain the origin and logic of form in the natural world.

1.1.3 Aristotle, Leonardo da Vinci, Galileo

Aristotle (384 - 322BC)

Whyte (1954) describes Aristotle as being the first organicist. Losee (1993) describes him as being the first philosopher of science. For these reasons, it seems natural to look to Aristotle to open the discourse on a philosophy of nature. His philosophy of nature grew out of a theological and metaphysical worldview, which was rooted in a doctrine of teleology, the doctrine of final causes. It comprised concepts of *agathon* (good), *beltiston* (best), *telos* (goal) and *hou heneka* (purpose), all referring to the ultimate purposiveness, the good and the omnipotence of God's intent. Chance and *automaton* (of itself) or *per accidente* (accidental causes) were not excluded, but they were considered by Aristotle to operate within teleological constraints. Wieland (1975) insists, *'that Aristotle's theory of teleology cannot be understood properly unless it is taken to presuppose his doctrine of chance.'* Losee (1993) explains that the final cause in Aristotle's doctrine of teleology relied on a process whereby sub-causes governed, and were governed by a final cause such that they *'presuppose that a future state of affairs determines the way in which a present state of affairs unfolds.'*

Aristotle's doctrine of teleology preached a viewpoint whose teachings emphasised the inherent meaningfulness and ultimate purposiveness of living nature. It beckoned to a First Cause or Grand Design. As such, Christian thinkers adopted its teachings.

The key to Aristotle's viewpoint is contained in his numerous texts, of which *Historia Animalium*[§] *Metaphysica, Physica, Politica, Poetica, Mechanica, Topica, Analytica Priora* and *Analytica Posteriora* are perhaps the most well known. Creswell (1862) refers to Aristotle's *Historia Animalium* as *'the most ancient and celebrated contribution which has come down to us; and it is hardly possible, when we consider the means of observation which were accessible at the time, to imagine a work of more accurate observation.'* The sheer scope of Aristotle's writings is a testimony of his all-

---

[§] An English translation of this text of Aristotle's was made in 1862 by Richard Creswell in ten volumes. D'Arcy Thompson also translated Aristotle's *Historia Animalium*. In the abridged version of Bonner, J.T, ed. (1961), Gould refers to D'Arcy's tranlation as the standard translation.

embracing concern with universal law. Of particular interest are the extensive series of mathematical discussions extracted by Heath (1949) from Aristotle's texts. Heath notes that Aristotle names mathematics as one of three 'theoretical sciences', the other two being theology (or first philosophy) and the philosophy of nature (or physics).

Aristotle, like other great philosophers, having formulated a philosophy of nature, used mathematics as a method of illustrating scientific method. His mode of proof was a cyclical process of induction-deduction. He made induction from observation, which led to other deductions and these deductions themselves could be verified through further observations. Aristotle's philosophy of nature contrasted with Pythagoras (570 - 510BC), Plato (428 - 348BC) and Euclid (300 - 260BC) who believed that processes and forms in nature could be explained by mathematical relationships or geometrical harmony.

Aristotle describes the practice of Art as a means to gain an understanding of nature. Art, he believed, either imitated nature or brought to completion a work which nature could not complete and had left aside. Art was a way of interpreting nature. Science, on the other hand was the product of a holistic quest for knowledge for its own sake, of the universal rather than of a part, and of the cause. The questions about the form of nature and about its cause were the aims of science to answer.

Wieland (1975) describes Aristotle's philosophy as having obstructed the progress of empirical science due to its rather dogmatic acceptance of an omnipotent God figure. He argues that scientists regarded the initial assumption of a supernatural force as an obstacle that prevented them from indulging in further investigation. Aristotle had rejected the theory of classical atomism, founded by Democritus (460 - 370BC) and Leucippus (450 - 420BC), and had instead accounted for natural phenomena through the action of a final cause. A decline in Aristotelian thought led to a gradual rise in scientific enquiry despite attempts by scholars like Leonardo who found merits in, and sought to pursue both channels of enquiry.

Leonardo da Vinci (1425 - 1519)

Leonardo was the universal scholar of the Italian renaissance. He expressed his endless enthusiasm and intellectual hunger as an engineer, scientist, anatomist, architect, philosopher, sculptor, artist, poet and writer. In his career as military engineer and architect

in service at the court of Lodovico Sforza of Milan, Leonardo had been pre-occupied with both physics and anatomy as records of his *Notebooks* (1490 - 1503) show. Brodownski (1976) summarises Leonardo's energy as having been '*dominated by two subjects. One - the internal mechanism of machines: wheels, pulleys, ratchets, and the arrangements which make a machine carry out a sequence of operations, step by step, automatically. The other - structural anatomy: the co-ordinated arrangement of bones and muscles which enables the animal body to move and act as a unity.*'

Leonardo epitomised the era in which there was no reason to be confined by the limits of philosophy, theology, art or science as separate thought processes. It was the era in which scholars believed that the answers lay in the whole of our existence and that knowledge about *all* aspects of human nature would provide the key to unlock the mystery of life.

Leonardo focused his energy on extracting examples from nature's forms and processes, using the understanding derived from these studies as creative tools for his artistic and scientific explorations.

Leonardo's epoch symbolised a time of reconciliation between Aristotelian teleology and the age of scientific discovery and empirical science.

Galileo (1564 - 1642)

As more and more questions were raised that could no longer be answered by untested faith in nature, enquiry was driven to the eventual abandonment of a universal teleological order, and to the birth of the era of modern science in the seventeenth century.

According to Losee (1993) rejuvenated theories of atomism that postulated that '*qualitative changes at the macroscopic level were attributable to quantitative changes at the atomic level*' heralded this departure from the traditional tenets of natural philosophy and the belief in the ultimate purposiveness of nature, to the tradition of empirical enquiry pioneered by Galileo and Descartes (1596 - 1650).

The object in science became the quest to find universal laws that would account for the processes that governed the anatomy of organic and inorganic matter, and which would enable the progress of science. This new goal of science contrasted with Aristotle's intent, which had insisted on making inductions from observing causes rather than establishing

laws through testing. Galileo found shortcomings in Aristotelian teleology and so reverted to the astronomical theories of Copernicus (1473 - 1543).

Galileo was the chief opponent to Aristotelian physics. He regarded Aristotle's philosophy of nature as being unscientific, partly due to his experience with false practitioners of Aristotelianism. In his *Dialogue Concerning the Two Chief World Systems* (1632) and later in his *Dialogues Concerning Two New Sciences* (1638), he set about to rigorously examine the concept of Aristotelian geocentricism as against the Copernican heliocentric system which was consistent with Pythagoras (570 - 510BC) and Kepler (1571 - 1630). He found favour with the Copernican system, which was in conflict with the views of the church. As a result of his agreement with and exposition of Copernican astronomy Galileo faced the Roman Catholic Inquisition of 1633.

Francis Bacon (1561 - 1626) is regarded as having taken up Galileo's banner in the reform and advancement of scientific method. Galileo was convinced that the secrets of the universe lay in the physical forms of things and that the discovery of nature's geometric rules would provide the key to its construction.

In Il Saggiatore (1623), he writes, *'Philosophy is written in this great book - I am speaking of the Universe - which is constantly offered for our contemplation, but which cannot be read until we have learnt its language and have become familiar with the characters in which it is written. It is written in the language of mathematics, and its characters are triangles, circles and other geometric forms, without which it is humanly impossible to understand a single word of it; without which one wanders in vain across a dark labyrinth'.*[§]

The age of analytical science pioneered by Galileo increased the scope of natural philosophy, and nurtured the age of scientific discovery, giving rise to many new sciences, amongst them physics and astronomy, which could now question, examine and test nature. Scientists cleaved unto physics, the science of non-living matter and biology, the science of living matter. In the 18th & 19th centuries, subjects such as morphology and comparative anatomy followed.

[§] Mandelbrot quoting Galileo in Mandelbrot, B., 'Fractal Landscapes', 1995.

Newton (1642 - 1727) belonged to the tradition of scientists who like Galileo concentrated their efforts on the physics of matter. We shall return to Newton later to discuss the significance of Newton's mathematical contributions in which he made it possible to describe irregular curvature.

But now we turn to the morphologists of the nineteenth century who began a methodical study of the forms and functions of living nature, inspiring the writings of Ruskin and the work of architects such as Morris, Viollet, Horta, Sullivan, Wright and Gaudí.

1.1.4 Vitruvius and Alberti

Worth a mention also, are the contributions of classical organicists, such as Vitruvius and Alberti, whose perceived rules of nature were reflected in the architecture of Ancient Greece and Rome. Classical precedent is apparent in the use of the lotus leaf and in other organic motifs in Ancient Egyptian monuments, as well as in the use of the acanthus plant in Ancient Greek architecture and in the use of flowers, plants and chimera in Gothic architecture.

Vitruvian ideals focused on proportion and symmetry and inferred a connection to the human body as *the* model of organic unity worthy of emulation. Alberti in *De re aedificatoria* spoke of the fundamental and absolute rule in nature of *concinnitas* defined as the quality of unity, beauty or harmony associated with nature and he insisted that it should be the main aspiration of architecture.

These theories will not be elaborated upon here. For further information, the reader is referred to a translation by Morgan (1960) of Vitruvius's *Ten Books on Architecture*, a translation by Rykwert *et al* (1988) of Alberti's *De re aedificatoria*, and van Eck (1994) for a full and detailed account of organic traditions in the context of classical architecture. Most accounts of organicism in architectural theory and the effect of it on product and process in architecture have associations with rhetoric and poetry. Architecture joins other art forms, in particular painting, music, poetry and literature in becoming a vehicle through which feeling is expressed using the abstract instruments of metaphor and rhetoric.

1.2 Organicism and 19th Century Morphology

1.2.1 Goethe, St.Hilaire, Pettigrew, Darwin

Goethe, of the German School of transcendental zoology or *Naturphilosophie* was one of the first to use the term 'morphology', as recorded in his essay of 1817 entitled, *Zur Naturwissenschaft uber-haupt, besonders zur Morphologie* (see Steadman, 1979). Goethe's work encouraged the growth of morphology as a branch of science whose main roots had come from the rapidly growing discipline of natural history.

In France, this discipline had been created out of the work of Georges Cuvier, the then Director of the *Muséum d'Histoire Naturelle* or *Jardin des Plantes*, as it was more popularly referred to and out of the work of others like de Lamarck, Brongniart and Saint-Hilaire.

The object of early morphological study was to develop a classification system that would account for all inorganic and organic matter by recording and grading their types into some hierarchical order based on external appearance, a project which Aristotle himself had embarked upon in his *Historia Animalium.* Haeckel was an important figure in Darwinian natural history, who in his studies had made numerous recordings of types of microcellular organisms, largely of the marine environment which were published in his German text, *Kunstformen der Natur.*

Morphological study in the nineteenth century was criticised, being referred to as non-functional anatomy, since it concentrated on form rather than on an analysis of function. The science of comparative anatomy took over in which functional characteristics were sought to provide clues about the origin of species.

Pettigrew's 3-volume discourse on *Design in Nature* (Pettigrew (1908)) is an exposition of the nineteenth century argument that surrounded theories of evolution. In his argument Pettigrew traces the doctrine of the origin of the species not to Darwin and his precursors, but to Aristotle. He names Buffon, Georges Louis Leclerc (1707 - 1788) of the *Jardin du Roi* or *Jardin des Plantes* as the *modern scientist* of the origin of the species. Lamarck (1744 - 1829) later joined Buffon in his attempts to formulate a theory to account for the origin of the species. Lamarck, who coined the term *biology* is regarded by many as being the *proto-evolutionist,* the precursor to Darwin. Cuvier, also in support

of Darwinian notions of evolution, regarded differences between species as being the result of adaptation to specific environmental factors or *conditions of existence.* Pettigrew further announces that Goethe in Germany, Geoffroy Saint-Hilaire in France and Erasmus Darwin in England all arrived at similar conclusions about the origin of the species in the period around 1795. He then lists a great number of naturalists who all helped to constitute the great wave of evolutionary theory that was about to break. Amongst them were Alfred Russell Wallace (co-discoverer, with Darwin of natural selection), Patrick Matthew, Unger, Alton, Huxley (a great champion of Darwin), Herschel, the astronomer, and endless others. Darwin's *Origin of the Species by Natural Selection* was published in 1859.

Pettigrew, in the second volume of his text of 1908, opposes Darwinian evolutionary theory. He summarises Darwinian theory as follows: *'Given a living primordial germ, everything (plant and animal) proceeds therefrom by variation in the fulness of time.'*

His copious argument against it is based upon the following:

'How is the stage of finality reached, where stability of form and infertility occur, if species are manufactured from varieties? Without a First Cause there can be no beginning, no continuity, and no end. If changes and improvements, in the case of species, are possible up to a certain point, why do they not continue? If existing species are descended from other or older species, whence came the originals? If endless modification in endless time is required to make a species, and if there are sub-species and varieties and gradation, there is no halting-place between the point of departure and the point reached: the traces of the plan become indistinct, and classification impossible. In other words the division into classes, orders, sections, families, sub-families, genera and species disappear, and all that is left is an interminable catalogue of varieties from the monad to the man.' Why do *'the oldest fossil plants and animals have their representatives on the earth [still] at the present day?'*

He concludes that *'plants and animals have no power to vary their structure and perpetuate the variation. No plant or animal (man included) can of itself, and independently, add to or take from its structural and fundamental endowments. The*

variations and adaptations are to be traced to the intelligence which pervades and regulates the universe.'

'There is nothing in nature to countenance the doctrine of chance, of natural selection, of spontaneous generation, which ignores a Creator and proposes to dispense with a First Cause and Design.'

Pettigrew believed that the central flaw of Darwinist theory was that it does not account for the spiritual personality of man, nor did it account for the predominance of ordered processes.

According to Garnham (1992), in the nineteenth century, Oxford University, the seat of classics, philosophy and history, was strongly opposed to the rise of 'science' and any threats to theology. The attempts to reconcile science and theology were expressed in the formation of a school of natural science. Natural science was regarded as the only acceptable, uncontroversial science because it studied *God's* creations. The British Association meeting of 1860 in Oxford, at which Huxley won the argument against Bishop Samuel Willberforce concerning natural selection, highlighted the inevitable discrepancies between natural science and religion. The Oxford Museum building, designed by Deane and Woodward, and completed in 1861, was the first public architectural exposition of the increased awareness at Oxford of this new science. Statues of scientific thinkers such as Hippocrates, Aristotle, Euclid, Galileo, Bacon, Newton, Leibniz, Linnaeus, Cuvier, Darwin and the Danish physicist Oersted were intended to embellish the perimeter of the museum court. In the words of the delegates of the Oxford Museum from minutes of their meeting held on 29 June 1855, the statues represented the *'Founders and Improvers of Natural Knowledge.'*

The Oxford museum was amongst the first buildings to announce the arrival of a modern tradition in architecture that re-affirmed nature as a model for ornament and form, followed by the Natural History Museum, London built between 1872 and 1881. Sir Joseph Paxton's Crystal Palace of 1851, although of this era, differed somewhat, in that it drew inspiration from the structure of the *victoria regia* water lily, i.e. its ridge and furrow design, rather than through its use as applied ornament, as was the case with the previous two examples.

1.2.2 D'Arcy Thompson (1860 - 1948)

We turn now to a natural philosopher and morphologist of the late nineteenth and early twentieth century who had the rare gift of gathering together information from a variety of disciplines to produce a concise argument to account for the law of natural form. He used knowledge of physical law and the tools of mathematics to produce a theory of biology.

It is not clear where the boundaries lie between the range of subjects that constitute D'Arcy Thompson's outstanding contribution to art and science. He was offered a choice of professorships in classics, mathematics and zoology, which seem adequately to address the question about the domain in which his unique talents belong. He, like Creswell prepared a translation of Aristotle's *Historia animalium.* Almost all inter-disciplinary research work that applies the lessons of biology to other subject matter makes inevitable reference to D'Arcy Thompson's *On growth and form*, (abridged edition, edited by Bonner (1961)).

D'Arcy Thompson's philosophy evolved out of Aristotelian teleology. He placed great emphasis on the effects of physical force, which he attributed to a deity rather than to chance design. Although D'Arcy Thompson opposed Darwinian theories of evolution he acknowledged a degree of self-contained adaptation within groups of specie, but was strongly opposed to the concept of cross-species adaptation.

His multi-disciplinary work applied mathematical logic to the processes of growth and development in a variety of species, and this work formed the basis of current inter-disciplinary research where mathematics is used to achieve shaping analogous to natural form. His life was devoted to the mathematisation of form.

D'Arcy Thompson used conformal mapping, a method of performing Cartesian transformations to depict differential growth. He produced non-linear distortions, which enabled him to compare members of related species. Through this comparison he could predict adaptation, genetic variation and development in organic species. He is amongst those responsible for recognising that organic form obeys far more complex rules than those of elementary geometry.

Warner (1857), who preceded D'Arcy Thompson, had clearly also grasped the fundamentals of nature's irregularities, as evident in his text, *Studies in Organic*

Morphology. He made similar attempts to lay down mathematical rules of non-linear curvature. Both Warner and D'Arcy Thompson distinguished themselves from early morphologists by crossing the barriers of morphology as a self-contained discipline, and taking it across into applied science where mathematical tools are used as a means to capture and manipulate the language locked in natural forms.

More recently, Lord and Wilson (1984) have provided a unified approach to the mathematical description of form in a way that is amenable to creative disciplines. Their work sought to encourage the development of a science of morphology, with the specific intention of its application to design and engineering.

1.3 The Early Modern Tradition in Architecture

The next section considers the visual impact of the use of nature's forms and shapes, which became manifest in the work of late nineteenth century architects. This organic tradition grew out of the Arts and Crafts movement pioneered by Ruskin, Morris and Macintosh in England and developing into the *Art Nouveau* supported by Viollet and Guimard in France, and practised proliferously by Victor Horta, Henry Van de Velde and Paul Hankar in Belgium. It seemed that closely at the heel of this movement were Louis Sullivan (who had studied at the *Beaux Arts* in France) and Frank Lloyd Wright who cultivated the American organic tradition and Gaudí, the Catalan *Modernisme* of the late nineteenth century.

In the twentieth century Le Corbusier took his cue from nature in the shaping of his rich and multi-facetted contribution to art and architecture.

1.3.1 Morris, Horta, Gaudi, Sullivan, Le Corbusier, Wright

William Morris (1834 - 1896)

The culture of William Morris permeated society largely through the applied arts. He did not complete the course to take holy orders in 1853 at Exeter College, Oxford, and neither did he fully pursue his intention to become an architect, even after a brief period of apprenticeship at the office of G.E.Street in 1855. He had all the time during his theological studies at Oxford maintained a strong interest in engraving, literature, poetry, architecture and painting.

It was this immersion into the arts that encouraged him to abandon first his theological career, and later his architectural career in 1856 to pursue painting, and to join the Brotherhood, following the determined persuasion of Dante Gabriel Rossetti, the pre-Raphaelite painter.

Morris had formed his creative opinions amidst a background of engravings, mediaeval stained glass, poetry and literature, ancient monuments and mediaeval manuscripts from the Bodleian Library. Following his travels to France and Belgium in 1854, he had been intensely influenced by Gothic Church architecture. In England, he had been enraptured by the writings of John Ruskin, Tennyson, Keats and Dickens.

Morris worked as a poet, engraver, painter and furniture designer and is regarded as the reviver of the tradition of craftsmanship. His designs for tapestry, wallpaper, chintzes, carpets, tiles and stained glass rejected the industrially produced artefact. He fought to incorporate human sensibility into functional art by infusing it with nature and allegory for the enjoyment of all.

The task of furnishing his first marital home, *The Red House*, in Bexleyheath, which was designed by Philip Webb, provided an opportunity for Morris to express his zeal to create useful, beautiful objects. He formed an association with other artists and designers in 1861, under the name of Morris, Marshall, Faulkner and Company and their occupation was in the Fine Art of Painting, Carving, Furniture and the Metals.

Morris's primary sources of inspiration in the development of his design and art included, Gerard's *Herbal*, an illustrated history of plants, Lewis F. Day's, *Analysis of Pattern* (1886 – 7), and Owen Jones's, *Grammar of Ornament* (1856).

Morris's wallpaper designs are renowned the world over, and still leave a distinctive imprint on many interiors. The William Morris Gallery in Walthamstow holds the majority of his works, including manuscripts and books produced by his printing firm, the Kelmscott Press established in 1890, and a selection of mementoes, engravings, textiles and furniture produced by his design firm.

This archive, together with the collection in the *William Morris Room*, formerly *The Green Dining Room*, at the Victoria and Albert Museum and that of the City Art Gallery, Birmingham form a continuing source of interest to lovers of the Arts and Craft tradition.

As an advocate of beauty and usefulness, he is regarded as being one of the prime initiators of the Arts and Craft movement in England, the Art Nouveau in Europe and of modern design world-wide.

In Watkinson (1979), Van de Velde, who was a contemporary of Horta and practitioner of the Belgian Art Nouveau is quoted as saying that *'the seeds that fertilized our spirit, evoked our activities, and originated the complete renewal of ornamentation and form in the decorative arts, were undoubtedly the work and influence of John Ruskin and William Morris.'*

Typically, Morris's designs were intensely worked drawings, rich in their tribute to nature. He achieved this richness through the persistent application of flat floral pattern and dense foliation. His working drawing for *Jasmine* wallpaper (1872), a rarely seen original design, is illustrated in figure 1.3.1a. Rich engravings and drawings provided decorative material not only for textile, tapestry, chintz and carpet design, but also for his typography and calligraphy.

Although he contented himself with the applied arts as a source of embellishment to architectural interiors, he perhaps unwittingly, set up the course for the revival and adoption of the ideals of organic beauty and functionalism in the proceeding era of architecture.

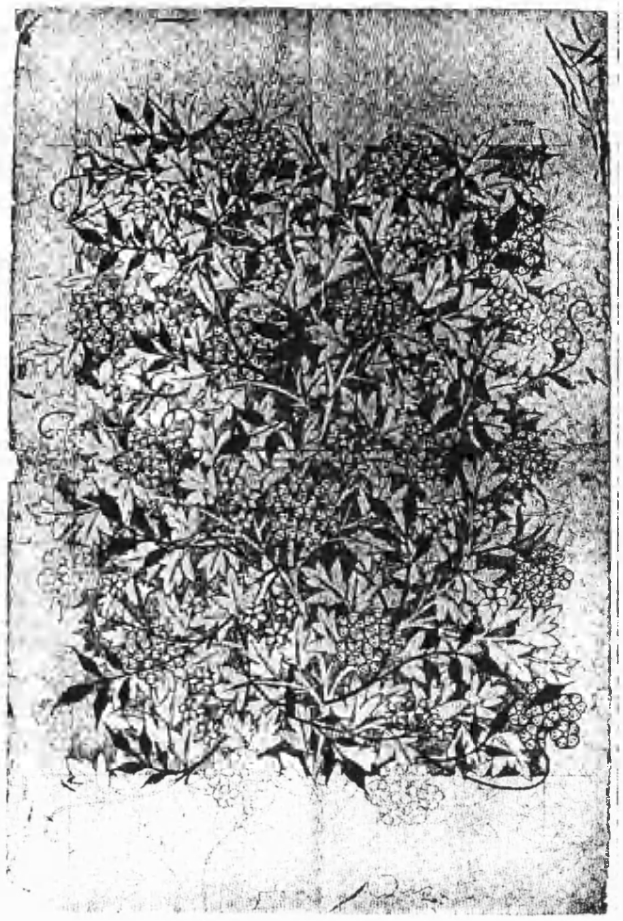


Figure 1.3.1a Jasmine wallpaper, working drawing by William Morris, 1872

Victor Horta (1861 - 1947)

Victor Horta pioneered the Belgian Art Nouveau. He borrowed the charm of the Parisian apartment block that had been created under the fervour and energy of late 19th century planning developments in Haussmann's Paris. According to *Archives d'Architecture Moderne* (1988), a larger planning budget in Paris permitted the development of grand apartment blocks built in the format of courtyard clusters along boulevards, whilst in Brussels, the pattern developed as a skimpier, leaner model of narrow ribbon apartments. It nevertheless gave Horta, Paul Hankar and Henry van de Velde, the opportunity to practice the *Art Nouveau,* a new blend of architecture which embellished the façade and interior with sinuous curves.

The movement provided a vehicle for socio-political change, and it was those who wanted that change, the new socialist intelligentsia, part of Horta's circle of friends, and the

Art Nouveau enthusiasts, who gave support to this new expressive architecture by becoming its willing clientele.

There is evidence to suggest a clear link between nineteenth century zoology and practitioners of the *Art Nouveau.* Haeckel's art forms of nature, which were first recorded in the Scientific reports of the voyages of HMS Challenger between 1880 and 1895 not only had an appeal to scientists. Haeckel himself sought to explore and encourage their aesthetic possibilities. The Dutch architect, H.P. Berlage owned a copy of Haeckel's catalogue, and based the designs of his light fittings for the Amsterdam Exchange on some of Haeckel's illustrations. Haeckel's drawings also had an intense influence on René Binet's nineteenth and early twentieth century designs for interiors and furniture, Louis Bonnier's pavilions for the 1900 Universal Exposition of Paris and Hector Guimard's designs for the Paris metro stations between 1899 and 1905. Binet, in a letter to Ernst Haeckel, admits that, *'everything from the general composition to the smallest details is inspired by your studies.'* The article concluded that, *'In a kind of artistic codicil to Darwinian theory, the plates* (presumably Haeckel's artistic plates, an example of which is given in figure 1.3.1b) *underwent an evolution of their own as turn-of-the-century European design marched off towards Art Nouveau.'*

A connection between art and morphological study at the time was clearly being established.

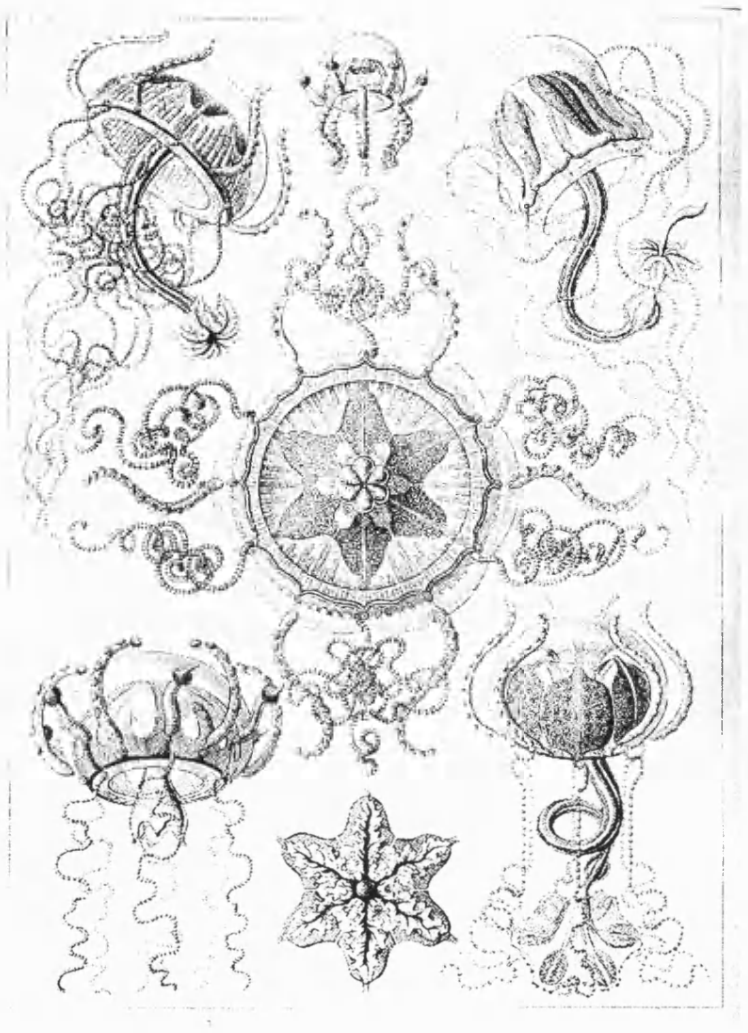


Figure 1.3.1b One of Haeckel's artistic plates showing Trachymedusae

Horta's first public statement of the *Art Nouveau* was embodied in the interior of the house he designed for M. Tassel, *Hôtel Tassel,* built between 1893 and 1895, located in Ixelles, Brussels. This commission gave him an unprecedented opportunity to liberate the language of the *Art Nouveau,* which he had previously merely experimented with at *Hôtel Autrique.* Figure 1.3.1c shows a mural he designed for one of the stair-cases of the *Hôtel Tassel.*

Probably, the most outstanding and most frequently visited of Horta's works is the studio he built for himself at 23, *rue* Americaine, Saint-Gilles in 1898 and lying adjacent to it, his house at no. 25. They both now comprise *Musée Horta.*

Both Dernie (1995), and Borsi & Portoghesi (1970), give good detailed accounts of Horta's contextual background and a full portfolio of the works he executed between

1889 and 1923. Whilst Borsi and Portoghesi (1970) infer a strong link between the curvilinear paths of swimming fish and the vocabulary of Art Nouveau, maintaining that this imagery presented Horta with a strong basis for his curved language, Dernie (1995) contests any obvious allusions to natural origins in Horta's pattern ornament of whiplash lines. Dernie instead attributes it to a more individual creative conviction that exists somewhere deep in the originality of Horta's mind. He regards any references in Horta's designs to nature as creating abstract symbols rather than making faithful tracings off an original. Horta's line, he says *'is individual and defies geometric construction.'* He however does describe Horta's details *'as being built out of forms of movement.' 'a movement composed of an endlessly curving line, which is Horta's personal attempt to depict the movement of Nature's creative essence.'*

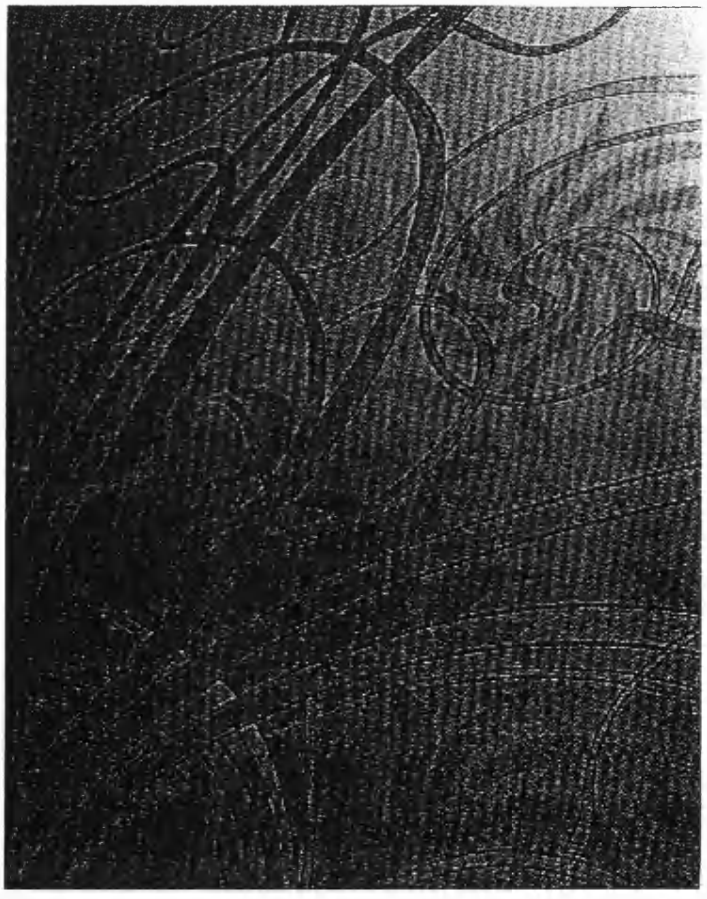


Figure 1.3.1c Detail of a ground floor stair mural in *Hôtel Tassel* by Horta, 1893 – 7.

Guimard also had the stamp of Art Nouveau, having been educated at the *Ecole des Arts Décoratifs* between 1882 and 1885 at the height of the movement towards floral pattern in design. He mimicked Viollet Le Duc and became greatly influenced by Horta after a visit he made to Belgium in the 1895. Guimard called nature *'a big book that we must look to for principles, which when found, have to be defined and applied by the human mind according to human needs.'* Dunster (1977) describes Guimard's wallpapers for Castel Béranger (Paris, 1894 – 1898) as having the substance of '*movement, ascending, waving stems and small motifs animated with inner life.'*

Figures 1.3.1d and 1.3.1e show Guimard's working sketches of a sign support and column base for the Paris Metro projects.

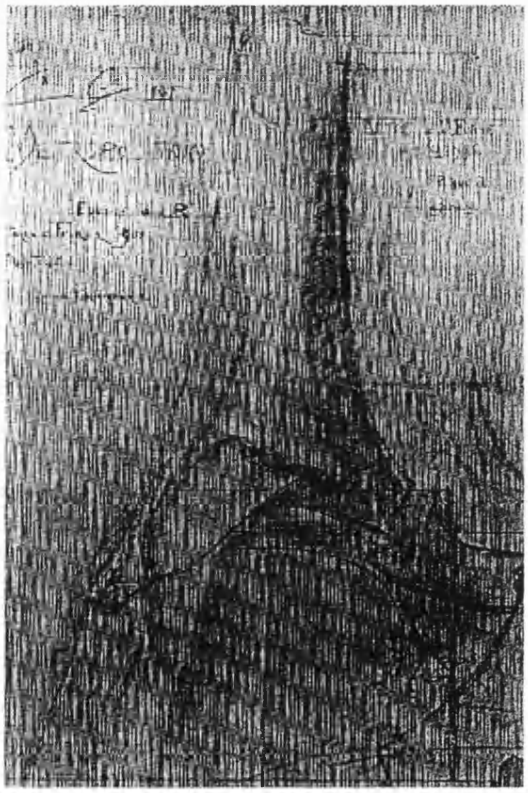


Figure 1.3.1d Hector Guimard's working sketch for a column base for the Paris Metro

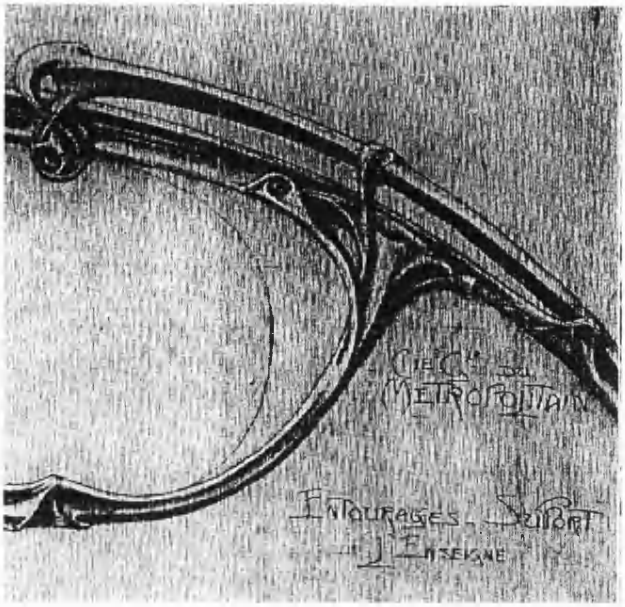


Figure 1.3.1e Hector Guimard's working sketch for a sign support, Paris Metro.

Gaudí (1852 - 1926)

Gaudí professed that *'originality is achieved by returning to origins.'* His work embodied an organicist aesthetic. According to Sterner (1985), Gaudí sought to fuse nature and geometry. He agreed with the aspirations of ancient thinkers and in their belief that nature and geometry were the two roots of all things.

For him structure and form were one, and he defined them through the use of parabolic and hyperbolic curves believing that they resulted from growth, and expressed the natural path of forces in equilibrium. He remodelled the Gothic arch by introducing a parabolic profile developed from hanging models[§], thus taking the sharp static point out of it. Sterner refers to Gaudí's use of the parabolic arch on *Palacio Güell* as the means to achieve a sense of motion, and to his use of curves on the façade of *Casa Milá* as *'sea frozen in motion.'*

His forms had a plasticity, which was derived from natural growth, and these forms he embellished with ornament and sculpted foliage. His sculptural buildings were executed

---

[§] The parabolic form results from a hanging thread loaded uniformly per unit horizontal length. A chain loaded uniformly per unit arc length hangs in a shape known as a catenary.

in iron, timber and stone; anything that was able to be carved and worked into a plastic state, taking a lead from his grandfather who had been a potter and from his father who had been a coppersmith. Sterner considered Gaudí to be the precursor in the trend toward plastic architecture, being heavily inspired by the Art Nouveau of Europe.

Gaudí was also intensely influenced by *the Gothic*. His forms draw heavily on plant and shell forms. Figure 1.3.1f shows a winding stair in one of the towers of Gaudí's *Sagrada Familia* church in Barcelona. Mainly due to the perspective created by the viewpoint of the photograph, the stair appears to be a near replica of C. Bergeau's sketch of *Heteroceras emerici,* a cretaceous shell drawn for Pettigrew shown in figure 1.3.1g.

Figure 1.3.1f Helical stair, *La Sagrada Familia*

Figure 1.3.1g Cretaceous shell

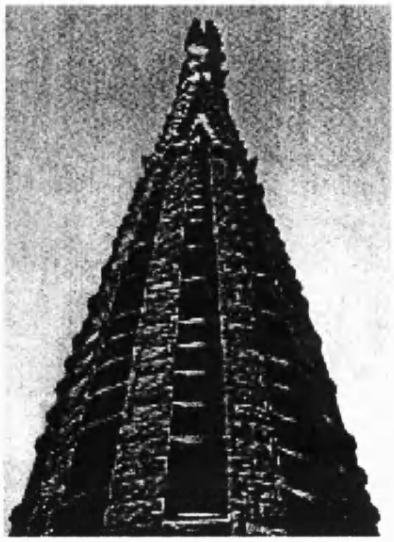


Figure 1.3.1h *La Sagrada Familia,* stone tower **Figure 1.3.1i** Tower finial

A view of one of the towers of Gaudí's still incomplete, but most famous work, the *Sagrada Familia*, is shown in figure 1.3.1h, and a detail of Gaudí's trademark gorgon-like finial at the top of the tower, is shown in figure 1.3.1i. Gaudí was exposed to the floral pattern work of William Morris and to the writings of Viollet-Le-Duc.

Louis H. Sullivan (1856 – 1924)

Louis Sullivan ascribed a mystical quality to the relationship between man, geometry and nature. In his book entitled, *'A System of Architectural Ornament according with a Philosophy of Man's powers'*, Sullivan introduces a theory concerning the creative nature of man's powers and then proceeds to show how they are implemented in design. His designs have poetic undertones, and pleasure can be derived from them as items of beauty in their own right, whether or not they are put to functional purpose.

Sullivan's argument is fundamentally organicist. He describes design as originating from a seed, which should be nurtured to encourage it to germinate and blossom into a mature plant.

Sullivan constructed highly ornamental drawings, which were worked up to a great intensity by the persistent manipulation, redrawing and re-interpretation of simple geometrical forms. He referred to geometrical forms as *'containers of energy'*, gradually evolving them into plant-like compositions – a process that one might compare with morphogenesis in nature.

Each stage in the development of his motif matured into an increasingly more complicated version of the original pattern. He infused inorganic motif with life borrowed from organic form.

The intensely worked drawings were used as friezes, motifs and decorative borders around the cornices and archways of his buildings. His decorative relief work embellished the flat surfaces of his buildings, which maintained a certain austerity; in Sullivan's eyes, the Richardsonian stoicism of his era, and floral motif were not in conflict with one another.

Figure 1.3.1j shows an example of Sullivan's technique of developing dense ornament from a 'seed' or 'germ', and figure 1.3.1k shows the façade detail of his National Farmers Bank of 1906 in Owotanna, Minnesota.

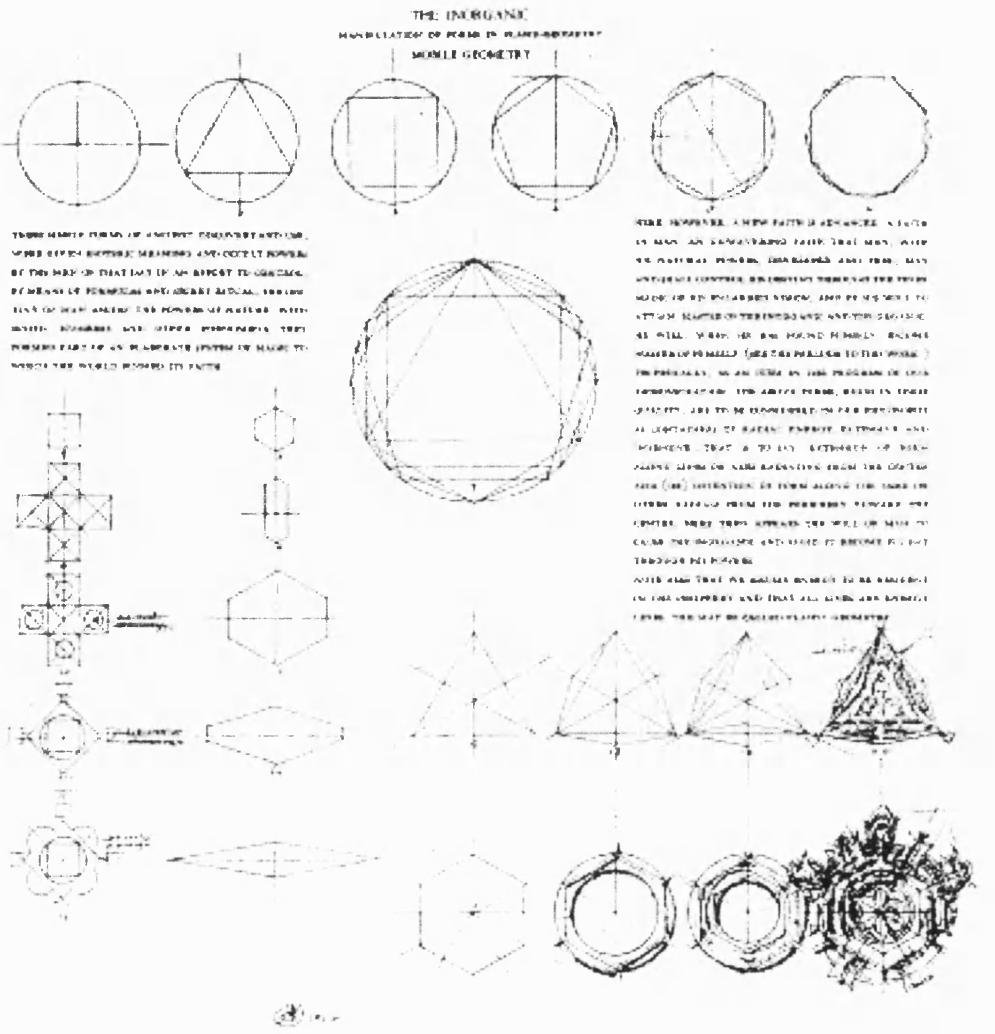


Figure 1.3.1j Sullivan's manipulation of plane-geometry

Sullivan's organicist influences have their root in the *Beaux Arts* tradition; he attended the *Ecole des Beaux Arts* in Paris between 1874 and 1876, where he was exposed to Viollet-le-Duc and to other pioneers of the emergent *Art Nouveau* movement of the late nineteenth century. He had earlier associations with Frank Furness in 1873, who himself had also used carved motif, floral design and leaf pattern. He drew also on the botanical archive in Asa Gray's *School and Field Book of Botany*, as well as on the eclectic pattern studies of Christopher Dresser and on the plant form studies of Ruprich-Robert.

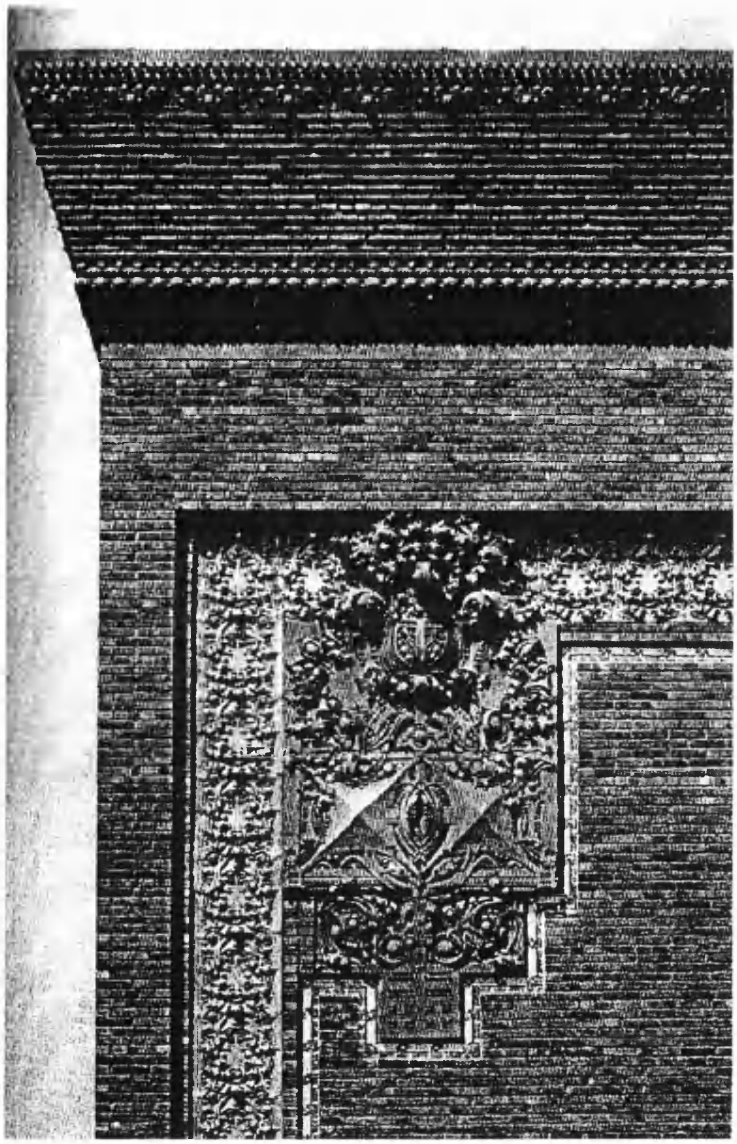


Figure 1.3.1k Façade detail of the National Farmer's Bank, Owotanna, Minnesota, by Sullivan, 1906

Le Corbusier (1887 – 1965)

Nature was a driving force in the development of Le Corbusier's thinking in art and architecture. His conceptual roots were rife with analogies of nature. He was influenced by L'Eplattenier, who believed that nature should be a primary source for the decorative artist, as did Ruskin from whom L'Eplattenier took his lead.

Le Corbusier had been making sketches from nature whilst a student at the *Ecole d'Art* in La-Chaux-de-Fonds where he studied engraving, design and painting. Von Moos

(1979) considered his training in geometrical abstraction at the art school to have the object of seeking *'the common ground where nature and mathematics meet.'*

Brooks (1987) and (1997) feature rare sketches from Le Corbusier's oeuvre showing the anatomy of a bird sketched by him in about 1904 and various interpretations of the cones and branches of spruce trees dated 1903, illustrated in figures 1.3.1l and m. Corbusier's sketchbooks feature several sketches of shells made by him from nature, two of which are shown in figure 1.3.1n.

In the years between 1917 and 1927, Le Corbusier underwent a change in his interpretation of nature, which in turn had an effect on his architecture. Becoming increasingly interested in developing purer forms, which established different relationships with nature, his architecture gradually evolved into cube compositions, which announced the arrival of *L'Esprit Nouveau* and the machine aesthetic.

In connection with Le Corbusier's emerging interests in industrialised production and simpler forms, he developed *Modulor* in 1946, shown in figure 1.3.1p, which was a culmination of efforts that had begun in the early 1920's to introduce intimate connections between the human proportion and machines for living.

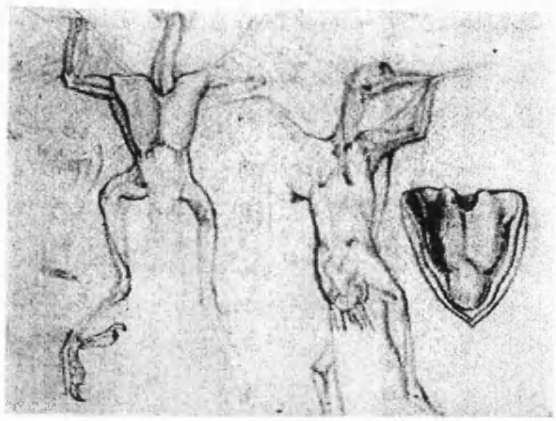


Figure 1.3.1l Anatomical sketch by Le Corbusier of a bird, 1904.

Figure 1.3.1m Sketches of spruce cones by Le Corbusier, 1903.

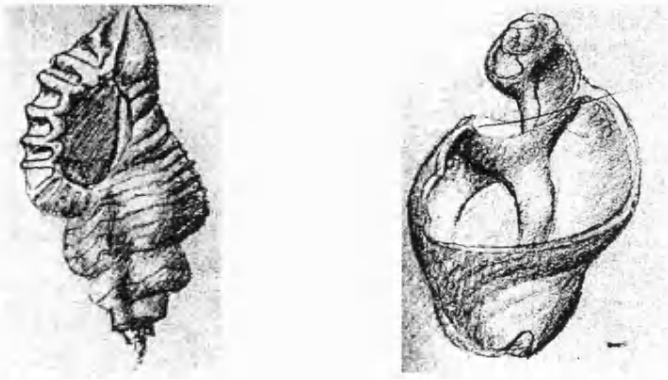


Figure 1.3.1n Shells drawn from nature by Le Corbusier

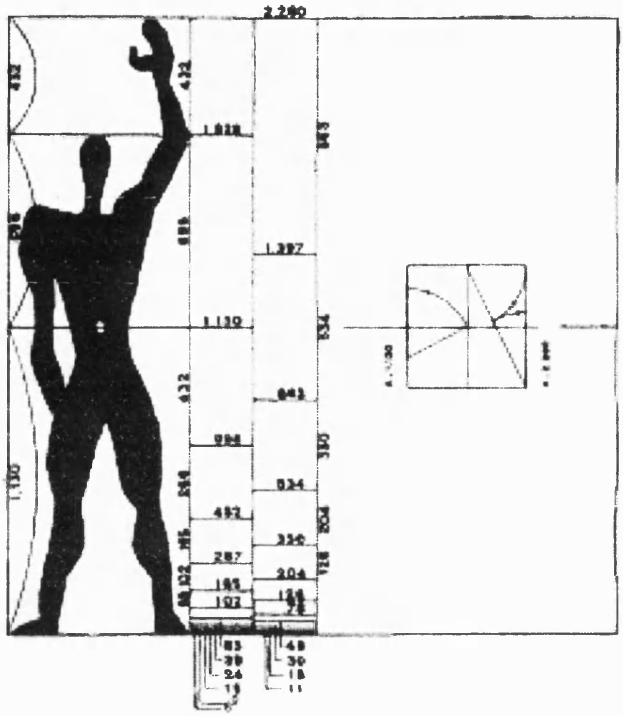


Figure 1.3.p The *Modulor* figure by Le Corbusier, 1946

Modulor provided Le Corbusier with an opportunity to combine nature and mathematics. He sought to rationalise the production of components for use in buildings and to standardise the dimensions of the fabric of his buildings. For this proportioning system, he reverted to the Ancient Greek idea of the Golden Section, which created a system of related proportions whilst preserving infinite variety within a pre-determined framework. He believed the proportions set up in the *Modulor* would lead to harmonious compositions which were modular without being repetitive, in a true ode to nature. Von Moos (1979) called *Modulor*, *'an organic numerical scale'* and Rudolf Arnheim referred to it as having the conceptual role of *'a Romantic variation of the Pythagorean philosophy.'*

Following the development of the machine aesthetic, Le Corbusier continued to explore nature through sketches, in particular the morphology of fossils, shells and bones and these sketches informed his sculptures and paintings of the late 1920's, as he began to make a departure from Purism. He called these sketches *objets à réaction poétique*

(objects which evoke emotion) – which Pauly (1997) refers to as being '*a formal repertoire that he drew on from the end of the Twenties for the basis of his design and pictorial research.*' Le Corbusier's still life compositions began to feature shells and other organic forms alongside inanimate objects.

Le Corbusier returned to an organic phase in the post-war period and this is clearly expressed in both his architecture and in his art.

The Jaoul Houses in France built between 1954 and 1956 re-introduce vernacular elements such as the barrel vault, suggesting a return to natural cave-like origins. During this period of reversion by Le Corbusier to an organic tradition, he also undertook the commission for the chapel of Notre Dame du Haut at Ronchamp illustrated in figures 1.3.1q and 1.3.1r.

Several sources have been cited as the inspiration for the roof of this chapel, amongst which is the suggestion that the offset roof floating over the pristine white walls recalls both the colouring and profile of a nun's headdress with hands clasped in prayer. Le Corbusier himself gives a more accurate analogy, as records from *'Création Ronchamp'* at the *Fondation Le Corbusier* Archives suggest. *'Thick walls and a crab's shell to give curves to a static plan. I'll provide the crab's shell'*, he says, referring to one he had picked up on Long Island beach whilst in New York in 1946.

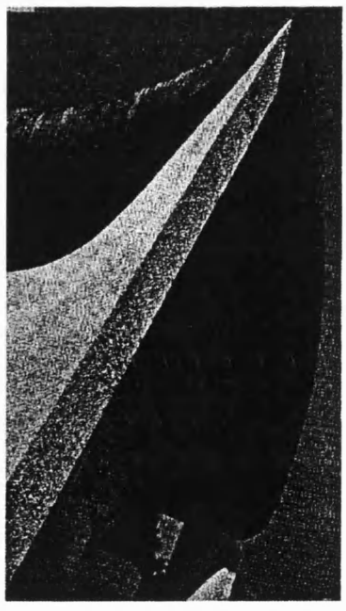


Figure 1.3.1q Notre Dame du Haut, Ronchamp, view from the south-east,

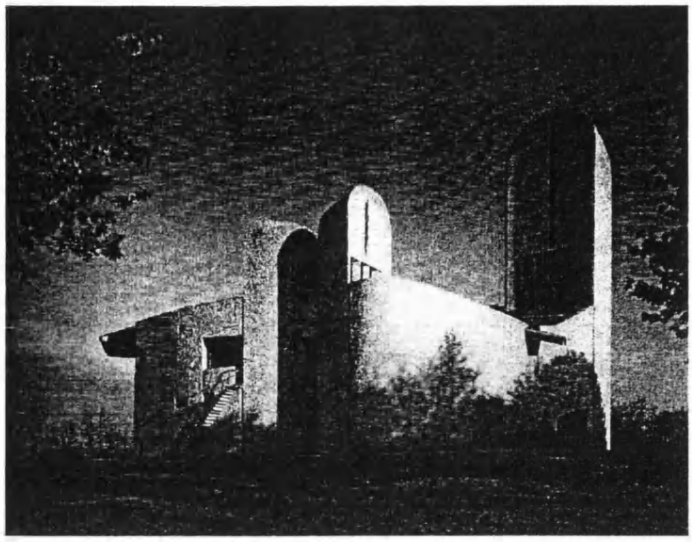


Figure 1.3.1r Notre Dame du Haut, Ronchamp, view from the north

One of Le Corbusier's unexecuted works was the *Mundaneum* (World Centre of Cultural Documentation) project for Geneva in 1928. The proposed museum building has a rectilinear spiral plan form, which is extruded vertically into a ziggurat. This spiral model was repeated in his design for the *Museum of Unlimited Growth* of 1939, intended for a Paris suburb. Le Corbusier had a fascination for spirals believing that they followed '*natural laws of growth, laws which underlie all manifestations of organic life.*'

Sketches of both these projects appear in Brooks (1993), showing how their geometries, albeit, rectilinear rely wholly on the infinite, spiralling of shell morphology. Le Corbusier travelled full circle in his creative journey with nature.

Von Moos (1979) made this observation about Le Corbusier's thoughts concerning the interconnectedness between geometry, nature and mathematics, that *'on the formal level, geometry frequently serves not only as an antithesis to nature but also as the mediator by which nature can be extended into the man-made environment. On the conceptual level, geometry, and mathematics in general, provide the structure through which nature, and the cosmos, can be understood and organized. To discover nature with the help of geometry and to use geometry as a cabalistic key – not only to an intellectual understanding, but also to a pantheistic experience of nature: these were the terms of Le Corbusier's beginnings.'*

He concludes by saying that, *'The Modulor brings them together into a system.'*

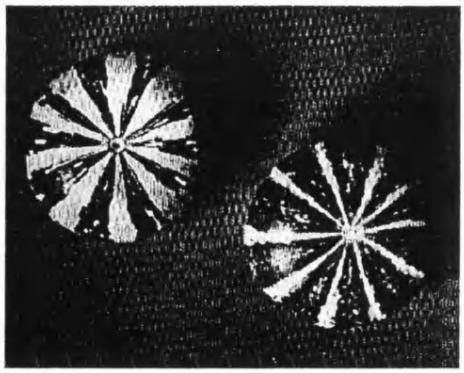
Frank Lloyd Wright (1867 – 1959)

Wright was chief of the American organic tradition and a pupil of Sullivan's. His early formation and exposure to the philosophy of the Friedrich Froebel Kindergarten system was responsible, to a great degree for his style of organic expression. Froebel's method used cubes, pyramids and other regular solids as a basis for teaching children of different ages about pattern and composition.

The Froebel philosophy was derived from uniformity and unity in nature, and much of its example was taken from the inorganic structures of crystals, rather than from living organisms. Froebel's approach evolved out of an analytical interpretation of the natural world, extracting from it a system of ordering principles and geometrical parts that could be built up into a structured and coherent whole.

In design, Wright took the Froebel inheritance with him and the result was architecture in which references to nature were structured, indirect and abstract.

Wright used stained glass in sharply defined geometric patterns, and the results were carefully abstracted reflections of nature, rather than mirror images. His designs captured the delicate silhouettes of tree branches and the fine tracery of winged insects. Figures 1.3.1s shows the lamp he designed for Dana House in Illinois, which is inspired by the fruit pods shown in figure 1.3.1r.



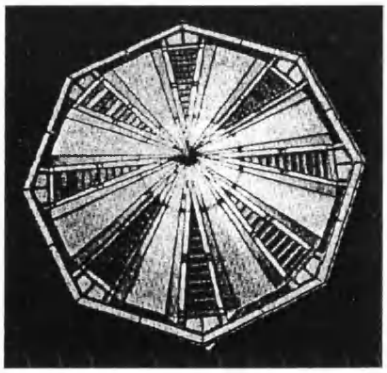


Figure 1.3.1s Wright's inspiration - Fruit pods **Figure 1.3.1t** Table lamp for Dana House

Wright's mission during his period of employment at Sullivan's Chicago office between 1888 and 1893 was in the strict geometric interpretation of Sullivan's organic

line. Where Sullivan used curling motif, Wright translated it into abstract geometry, space and materials, yet quite remarkably, avoided any conflict with Sullivan's intentions. He balanced the two extremes of Froebel's geometric order with Sullivan's highly developed ornamental compositions.

Following the establishment of his practice in 1893 with Cecil Corwin, themes of nature were echoed throughout Wright's design vocabulary. He used natural materials and he portrayed nature's infinite quality by incorporating freedom and openness into his interior planning.

The perfection of his creative methods and ordering principles was achieved in the development of his 'prairie style' houses, which were characterised by their delicate communion with nature. They absorbed and echoed their sites. They stood as abstractions of nature's themes and captured its essence through subtle relationships rather than through direct imitation. They were laid out expansively, like nature, free and all pervading.

A particularly successful attempt of a fusion of landscape with architecture is Wright's studio of 1938 at Taliesin West, near Phoenix in Arizona, whose plinth and main structure is built from desert concrete, a roughly shuttered concrete of boulders and stone. The resulting structure appears to grow out of its surroundings, despite its sharp corners and hard line. A detail of it is shown in figure 1.3.1u.

Wright's Kauffmann house of 1936 in Bear Run, Pennsylvania, illustrated in figure 1.3.1v and more intimately referred to as *Fallingwater,* is of the same genre. Despite its hard line and detachment from the rugged surroundings, its hovering cantilevers could not be more appropriate to the site.

Wright seldom introduced curves into his designs; he never sought to imitate the forms of nature. The Guggenheim Museum in New York is one of the few examples where Wright abandoned the emblematic tools of his trade, the T-square and triangle, for the ascending spiral.

Figure 1.3.1u Detail of desert concrete and redwood trusses, Talisien West, 1938.

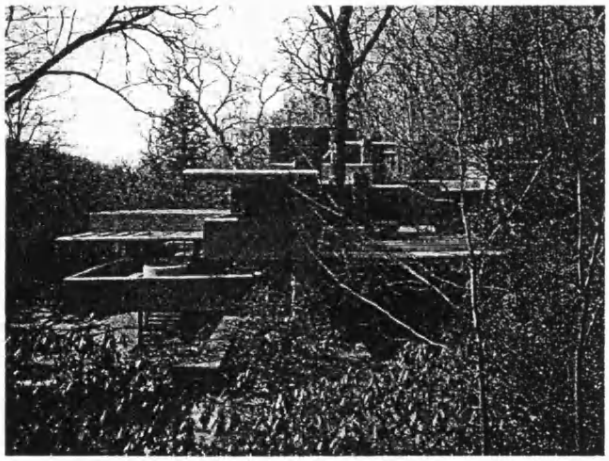


Figure 1.3.1v *Fallingwater*, 1936, entrance side from the east

During a visit he made to Japan in 1906, Wright cultivated Japanese principles of abstract design, which were in harmony with nature, but did not imitate it. Japanese influences reinforced his own ideas. He reconciled his growing awareness of nature through abstract representation.

Zevi (1950) observes that Wright's buildings are *'real and intensely personal; inseparable from the life which goes on within them and the life of nature that goes on outside them', and he regards them as being 'based on a social rather than on a figurative idea.'*

Wright's architecture is as much figurative as it is social, and it is in fact by virtue of its figurative ideals that it achieves the social ideal.

In Wright's own interpretation of his work, he remarks, *'I find it as a new sense of reality, an earnest life-long search for that thing growing out of the nature of the thing, not from anything applied to that thing from without.'*

1.4 The Sculptural Tradition in Engineering

1.4.1 Nervi, Torroja, Candela

Nervi, Torroja and Candela had much in common; all three had ambiguous identities with regard to their creative personalities; they swayed between the poles of architect and engineer. Of the three, Candela alone trained as an architect, absorbing into his training, an engineering sensibility, whilst Nervi and Torroja trained as engineers and absorbed into their training, architectural sensibilities. The resulting trio are the great shell engineers of architecture and sculptors of engineering.

Pier Luigi Nervi (1891 – 1979)

Nervi was amongst the leaders of the aesthetic tradition in reinforced concrete structures, which began with the discoveries of Smeaton and Monier in the early part of the nineteenth century and found its greater potential with Hennebique, Perret and Freyssinet in the 1890's. The tradition continued with French engineer, Robert Maillart, Swiss-born architect, Le Corbusier and Swiss bridge engineer, Christian Menn. The Spanish engineer-architect, Santiago Calatrava, continues the tradition today.

Nervi's concrete aesthetic had dynamism, clarity of form and a three-dimensional structural integrity.

The rhythmic harmony of his halls speaks at once of structure, and of architecture. Nervi's structural language has strong associations with ribbed shell forms; in particular, the precast dome on his arena of 1958 for *Palazzo dello Sport* in figure 1.4.1a, appears to be scaled up version of the costate cockle shown in figure 2.2b.

The dome is approximately one hundred metres in diameter and comes to rest on fan-shaped buttresses, which in turn are supported on inclined *ferro-cemento* columns. The total zone dedicated to receive the thrust of the dome roof is approximately twenty-five

metres, measured from the seating of the precast ribs at high level to the foundations of the columns below ground.

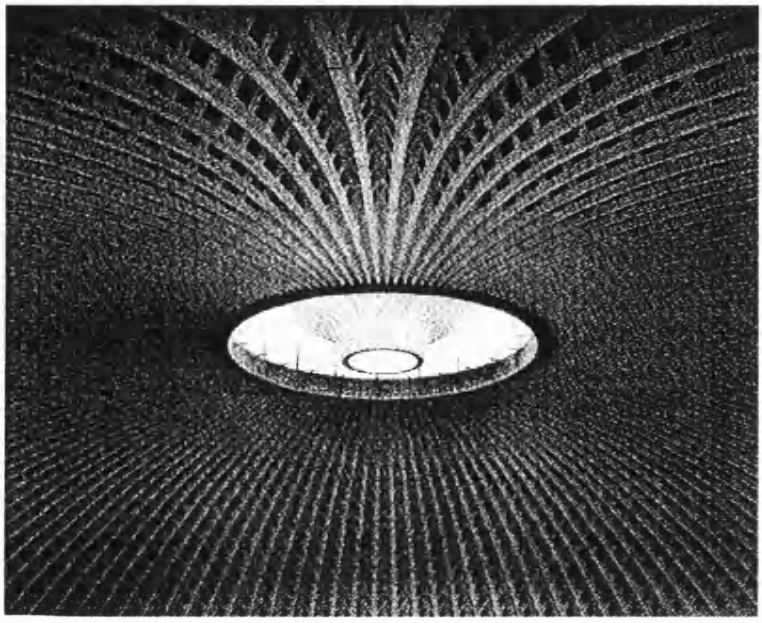


Figure 1.4.1a *Palazzo dello Sport*, Rome, 1958-60, interior view of dome, 330ft dia.

The 'Nervi system' for pre-casting concrete units led him to create vast vaulted structures for exposition halls, hangars, factories sports arenas, stations and warehouses. His invention of *ferro-cemento* led him to create forms of unparalleled plasticity. Huxtable (1960) describes the forms as,

'free, curving ribs or undulating corrugated slabs of superb structural efficiency, relating directly to the static forces of complex structures by directly following the main lines of stresses.'

Huxtable considers the network of beams on the soffit of the Gatti Wool factory to be a fulfilment by Nervi of Alberti's 15th century description of nerved netted vaults, and she quotes Alberti in saying that,

'in all Manner of Vaults, let them be of what Kind they will, we ought to imitate Nature, who, when she has knit the Bones, fastens the Flesh with Nerves, interweaving it everywhere with Ligatures, running in Breadth, Length, Height and circularly'.

She adds that Nervi was echoing Alberti's words in talking about the reinforcement of concrete structures when he said that,

'the pattern of steel should always have an aesthetic quality and give the impression of being a nervous system capable of bringing life to the dead mass of concrete.'

Figure 1.4.1b below, shows a fine example of Nervi's plasticity of form, in the use of warped pilotis for the proposed extension to the UNESCO headquarters in Paris.

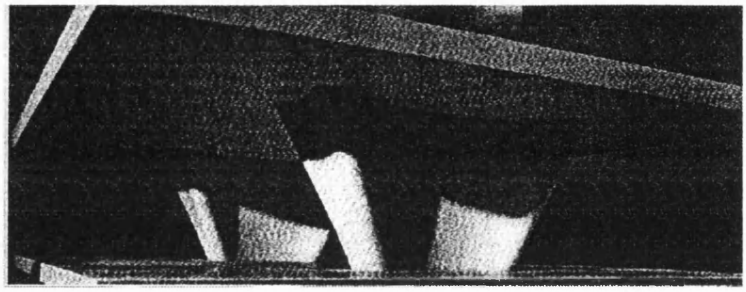


Figure 1.4.1b Nervi's study for concrete pilotis, proposed extension to UNESCO headquarters, Paris, 1958

Above all, Nervi believed that architecture should be a stable, unified, balanced, organism, capable of giving pleasure to the senses, and that its structure and form were to be created out of the laws of statics.

Eduardo Torroja y Miret (1899 - 1961)

Torroja trained as a structural engineer in Madrid at the *Escuela Especial de Ingenieros de Caminos, Canales y Puertos* between 1917and 1923. During his career as an engineer, which spanned over thirty-two years, he designed a large number of civil engineering structures. With little exception, his structures draw heavily on shell forms. His primary concern was with structural behaviour and efficiency, yet his structures have a natural grace and elegance. In 1959, he founded the International Association for Shell and Space Structures.

Torroja leaves no real clues in his writings as to whether he was directly inspired by natural shells or whether he was purely concerned with shell action in the structural sense,

for he did not often make any explicit references to nature. In the autobiography about his work, published in 1958, it is only in relation to his projects for churches that direct references to the natural shell are made.

The original design of his Chapel of the Ascension at Xerallo, and his church of Pont de Suert, at Lérida built in 1952 in collaboration with the architect J.R.Mijares, incorporate clusters of minor shells whose apices support a major shell to provide height and grace to the interior space. The Pont de Suert church in particular, has a classical plan, but Torroja superimposes a sculptural form onto it.

In his description of it he makes a specific statement about the allusion of its lobes to seashells, and he makes reference to the cross-sectional form of the cupola apse as being *'that of a logarithmic spiral, the pole of which lies on the contour of the opening of the nave into the apse.'* A similar form is adopted for the outdoor altar of the Sancti Spirit church of 1953. Figures 1.4.1c and 1.4.1d show working models of the lobes and cupola of the Pont de Suert church.

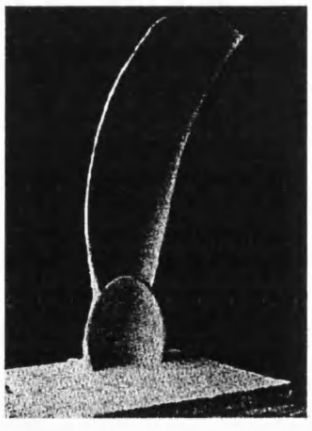


Figure 1.4.1c Detail of wall lobe

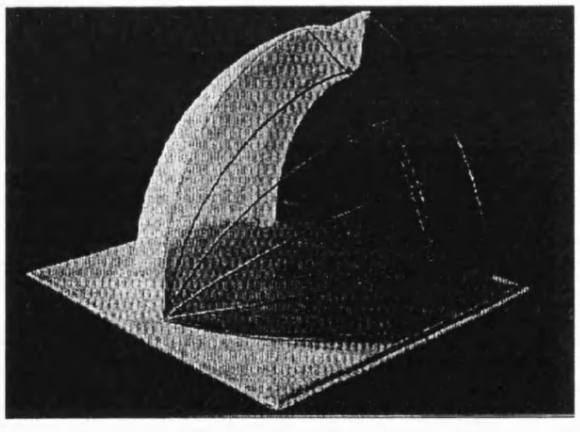


Figure 1.4.1d Detail of cupola apse

The elegance of Torroja's structures arises undoubtedly from the use of sweeping curves, which he took care to generate mathematically, and to prove structurally. In many cases, he maintained that curvilinear geometry lent itself well to bending moments acting on structures, with the aid of appropriately placed reinforcement. He echoes Maillart's words - 'force follows form'.

Felix Candela (1910 -1997)

Felix Candela was a near contemporary of Torroja. He designed and built in the generation preceding Calatrava, sharing with Torroja and Calatrava a Spanish origin and a passion for the hyperbolic parabola. He learnt the experience of shell analysis from Robert Maillart, and although Candela had trained as an architect, he developed an exceptional feel for the mechanics of elastic systems. He instructed himself and others about the principles of design for structurally indeterminate systems.

Candela realised the majority of his works in Mexico between 1940 and 1950, where he sought refuge during the war years. Mexico was an ideal environment for the development of his complex form language, because the labour and material associated with providing moulds for his structures was cheap and available. In addition, Mexico was fertile ground for large scale building work.

The distinctive parabolic profile of the roof of the Cosmic Ray Pavilion shown in figure 1.4.1e, which he designed in collaboration with Jorge Gonzalez, is one of Candela's most famous works.

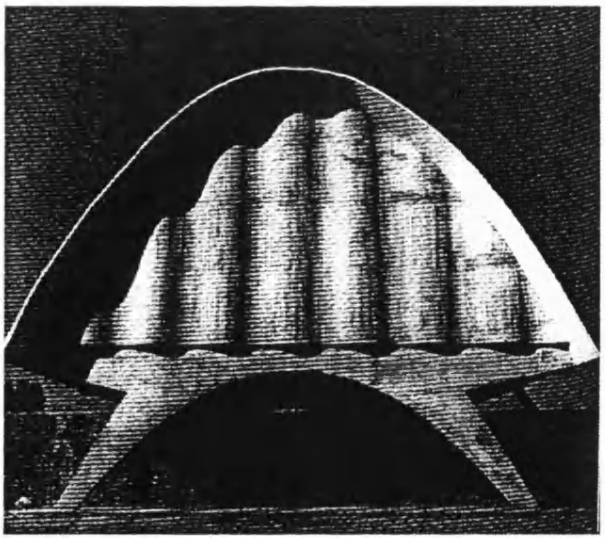


Figure 1.4.1e Candela's Cosmic Ray Pavilion, Ciudad University, Mexico, 1952

Candela worked as both architect and contractor, having formed a building-contracting firm in 1950, called, *Cubiertas ALA*, specialising in roof construction. Candela's main

concern was how to design, how to calculate and how to build shells, which almost always consisted only of a roof and supports. For Candela, shell architecture presented the opportunity to do away with walls; the roof was sufficiently plastic, and as a condition of its structural integrity, it relied on being warped and twisted.

He exploited the plasticity of concrete in order to create distinctive architecture. It may only have been coincidental to Candela that the natural shell existed. He seemed to be more concerned with abstract notions of the shell as a structural type and technique of building rather than as a technique of nature.

1.5 The Structuralist Tradition in Architecture

1.5.2 Fuller, Le Ricolais, Otto

Richard Buckminster Fuller (1895 -1983)

Buckminster Fuller, the self-styled, self-titled hero of architecture gained his proper position and status by lecturing to audiences in Schools of Architecture throughout the United States of America and abroad, without any formal training in either architecture or engineering.

Fuller's experience was gained largely as a result of his period with the US Navy during the First World War, and subsequently with his work in construction for the Stockade Building System in 1923. His experience with Stockade kindled an interest in prefabricated housing that embodied the qualities of both aircraft and automobile. A large number of his modified grain bins, the Dymaxion Deployment units, were adopted by the USAAF as accommodation for their aircrew.

It may seem a little difficult at first, to evaluate the intimate connection between Fuller and nature, given the industrial nature of his inventions. The key to Fuller's nature lies in his interpretation of geometry He follows the tradition of Pythagoras and Newton in his distillation of nature from universe into atom. He believed that energy patterns in nature could be expressed using geometric solids. He sought to minimise energy and maximise efficiency, which led to his maxims of *Synergy*, by which he meant synthesised energy, and *Dymaxion*, coined for him from his word trio, *dynamism*, *maximum* and *ions*, by the wordsmith Waldo Warren.

From nature, he learnt the fundamental geometry of closest packing, and through topological experiments in the close packing of spheres, he developed the geodesic dome. Fuller's geodesic domes are constructed from closely packed spheres by projecting the spheres onto triangular, pentagonal and hexagonal faces.

He discovered that close-packings could be developed from 12, 42, and 92 spheres, ad infinitum. He made the further observation that Uranium, the 92nd element in the atomic series, contains 146 neutrons, which equates to adding 12, 42 and 92. The carbon-60 molecule in figure 1.5.1c, a highly stable molecular state of carbon, is nick-named the *Buckyball*. Its official chemical name is the *BuckminsterFullerene*, after him. The physical structure of carbon-60 displays a closely packed arrangement of 12 pentagonal faces and 20 hexagonal faces, a configuration that typifies Fuller's geodesic domes, in particular the truncated icosahedron.

An ingenious example is shown in figure 1.5.1a of Fuller's foldable geodesic, which has nodes fitted with gas-pressured cylinders to drive the masts that deploy it.

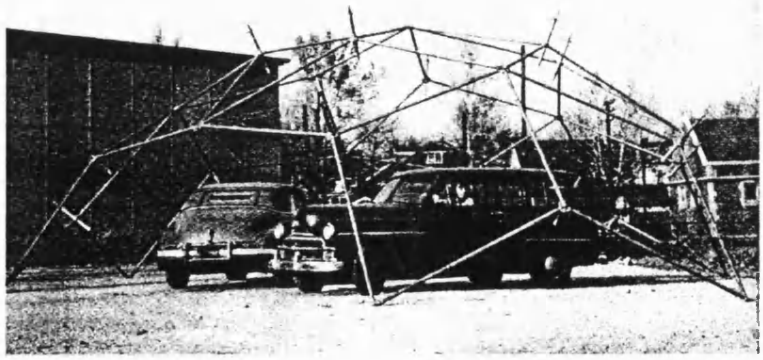


Figure 1.5.1a Fuller's foldable geodesic measuring 42ft, deployable in 45 seconds.

An infinite variety of polyhedral structures exist naturally in planktons such as *radiolaria*, *foraminifera*, *coccolithophores* and fossil coral, shown in figure 1.5.1b, as well as in the compound eyes of insects. Wester (1977) notes that *coccolithophores*, in particular, form their structures through an agglomeration of closely-packed buds or *coccoliths*, and attributes their non-oriented spherical and polyhedral geometry to the equal

distribution of water pressure surrounding them. In this regard, Fuller observes that it is a fundamental property of geodesic spheres to enclose maximum volume with the minimum surface area, thereby minimising energy. In nature, energy is dissipated evenly and in Fuller's geodesic structures, it is minimised evenly.

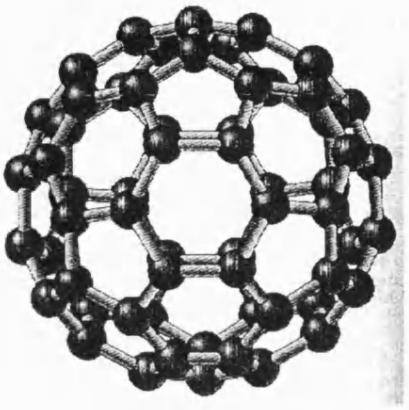


Figure 1.5.1b Full-grown mass of fossil coral Figure **1.5.1c** Atomic structure of *Buckyball*

Fuller's Dymaxion World Map, which earned him a US patent in 1946, practically eliminates the distortions that arise in standard map projection. He developed the map by projecting the grid of a spherical icosahedron onto a flat system of squares and triangles, an illustration of which is given in figure 1.5.1d.

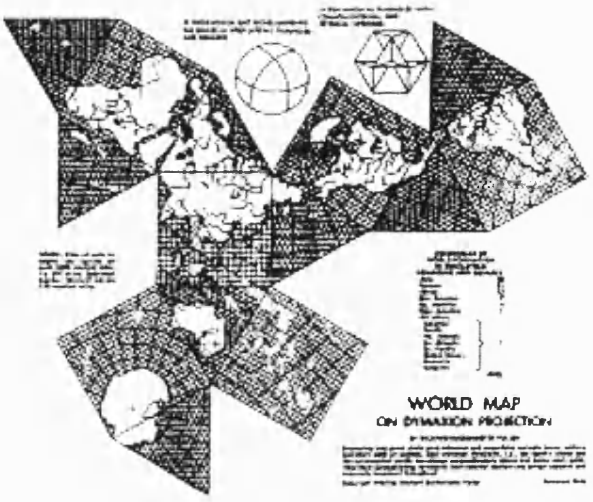


Figure 1.5.1d Buckminster Fuller's Dymaxion World map

Fuller believed that nature could not be partitioned into the separate subjects of chemistry and physics, and that the lessons of nature's geometry were worthy of emulation. Above all, Fuller was a cosmogonist who applied his thinking firstly to geometry and ultimately, to architecture.

Robert Le Ricolais (1894 - 1977)

Le Ricolais is perhaps the most underestimated figure in the search for the prodigy of nature as inspiration for design. His audience and readership are close to extinction and English versions of his writings produced during his long association with the Graduate School of Design in Pennsylvania are in limited circulation.

In the winter of 1998, the author was invited to write a review of an exhibition held at the Architectural Association to celebrate the work of Le Ricolais. The text of the review is reproduced here as it appears in *AA Files 39*, dated autumn 1999.

Robert Le Ricolais – Visions and Paradox

AA Exhibition Gallery 11 January – 5 February 1999

Emma Nsugbe and Chris Williams

Robert Le Ricolais was born at La Roche-sur-Yon, in western France, in 1894. His studies of physics and mathematics in Nantes were halted by the First World War, in which he was wounded. He was awarded the Military Cross and the Croix de Guerre, with two citations.

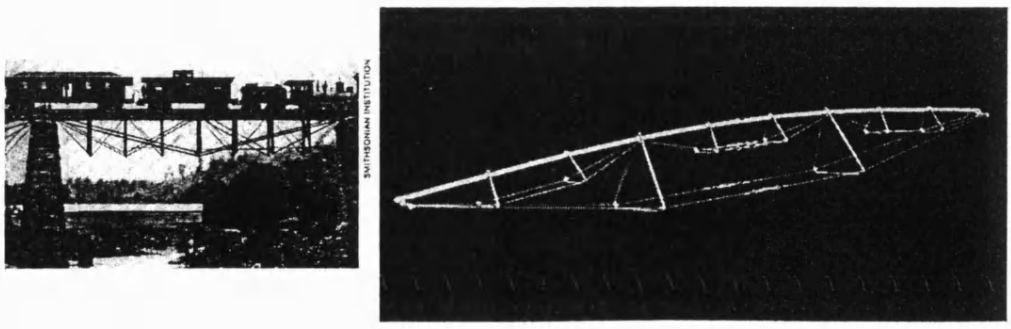
Between 1918 and 1943 Le Ricolais worked as a hydraulics engineer, becoming a director of the Société de l'Air Liquide in Nantes. It was during this period that he began to develop his ideas on cable structures, grid shells and space frames. Le Ricolais was awarded the Medal of the French Society of Civil Engineers for his article of 1935 entitled 'Les Tôles composées et leurs applications aux constructions métalliques légères' and for his subsequent work on corrugated stressed skins. In 1962 he was presented with the Grand Prix of the Cercle d'Études Architecturales de France for his two papers on space frames written between 1940 and 1941 and published in the *Annales des Ponts et Chaussées*.

In 1951 Le Ricolais moved to America in search of a more welcoming audience for his creative experiments in engineering structures. He taught in Pennsylvania between 1954 and 1976. After the death of Louis Kahn in 1974 he held the Paul Philippe Cret Chair in Architecture until shortly before his own death in 1977.

Le Ricolais's work has been most extensively published in France. In addition, a series of interviews with him were published in *VIA 2: Structures Implicit and Explicit*, a publication of the Graduate School of Fine Arts, University of Pennsylvania. Le Ricolais became a Fellow of the American Institute of Architects in 1973, which honoured him with a Research Medal in 1976. Apart from his work as an engineer he published poems and exhibited his constructivist air-brush paintings. The Museum of Fine Arts in Nantes holds many of his paintings.

Le Ricolais's vision and philosophy were as potent and original, perhaps even as visionary, as those of Richard Buckminster Fuller, his contemporary. The time is long overdue to celebrate his work and to bring it to the attention of a wider European audience. The Centre Pompidou in Paris included Le Ricolais in its exhibition *The Art of the Structural Engineer* in the summer of 1997. In October of the same year the cultural foundation COAM in Madrid mounted an exhibition devoted entirely to Le Ricolais, initiated and curated by Professor Peter McCleary of the University of Pennsylvania. In January 1999 this exhibition moved to the Architectural Association as *Robert Le Ricolais – Visions and Paradox*.

The exhibition consisted of models and drawings made by Le Ricolais and his students. There were pre-stressed cable beams and towers, grid-shells and cable-braced trusses. Particularly elegant was one of Le Ricolais's designs for a truss, which he called the Polyten truss. Dating from American railway engineering of the nineteenth century, and derived from the Fink truss, it is based on the idea of decomposing the



COLUMNS SUSPENDED IN THE AIR

Left: Railway bridge using the Fink truss, near Lynchburg, Virginia, 1875. Right: Le Ricolais's Polyten truss. Compression and tension elements decomposed.

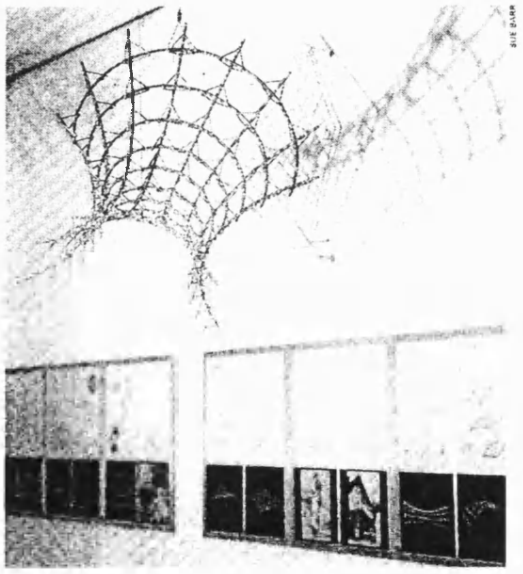
tension and compression elements.

We might ask why Le Ricolais is not more widely known, especially among architects who have embraced the structural aesthetic. Perhaps this is because engineering has no cult of the individual: who, except for *aficionados*, can name the designer of the Citroën DS or the Boeing 747?

Buckminster Fuller, who remained on the periphery of both architecture and engineering, suffered from the same uncertain status: he was not recognized during his lifetime. But he did achieve much greater recognition than Le Ricolais, and remains an icon of architectural vision. This may be due to Fuller's association with the US Army Corps, whom he persuaded to adopt his geodesic prototypes, dymaxion pods and airships. In contrast, most of Le Ricolais's structures, despite their unsurpassed elegance, remained unbuilt. He often mused over the lack of success of his proposals for radar dishes, trihex domes and transmission towers, and confessed to a certain lack of the business acumen and stamina required to find a client, convince them of the viability of a proposal, and have it built. Le Ricolais was more interested in research because it involved a longer-term exploration of a variety of solutions that could contradict one another or remain inconclusive.

Le Ricolais often compared his approach with that of Fuller. Despite an apparent similarity of purpose, Le Ricolais considered his ideals to be curiously in conflict with those of Fuller. He criticized Fuller's spherical geodesic domes, arguing that a sphere and a dome are not the same: 'The difference is only slight', he remarks, 'but, nevertheless, significant.' Le Ricolais was making an important point about the regularity and symmetry of geometry, which can often conflict with the subtly irregular shapes in nature that are created by natural forces.

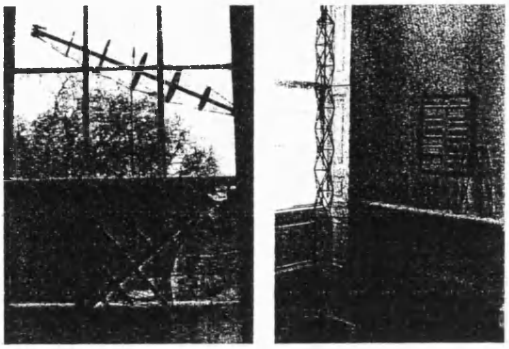
Le Ricolais applied his imagination to many concepts that we often take



View of the installation in the AA Exhibition Gallery

TRIAXIAL NETWORKS

Left: *Auloscena* radiolarian, from: E. Haeckel, *Kunstformen der Natur* (1904). Centre: 1:200 scale model of Le Ricolais's Trihex dome, with a span of 200 metres and a rise of 60 metres. A semi-regular tessellation of regular hexagons and triangles. Right: Le Ricolais's Starhex dome, similar in topology and span to the Trihex.



CABLE-BRACED TRUSSES AND COLUMNS

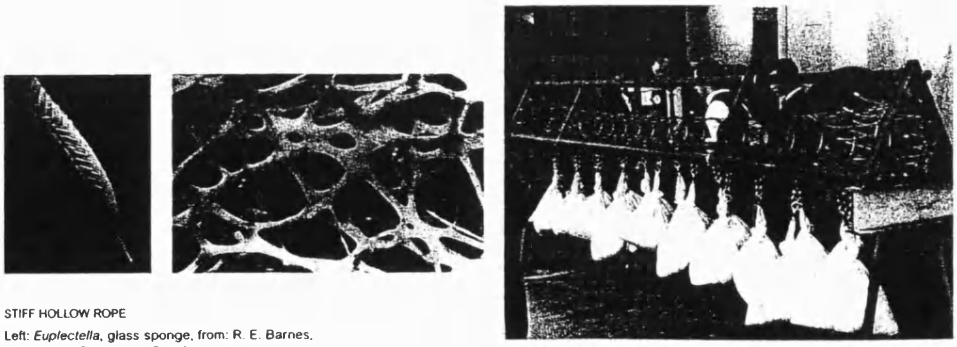
Left: View of the AA exhibition installation showing the Funicular Polygon of Revolution truss (above), and the Funicular Polygon of Revolution Pseudosphere of 1961–2 (below). Right: View of the exhibition showing, in the foreground, the Octen antenna, a tensioned steel structure of octahedral units pre-tensioned by steel cables and, in the background, Automorphic tubes T6 and T12, which demonstrate the principle that, with harmonic buckling, the tubes are capable of withstanding a higher axial force – the beauty of failure.

for granted. With regard to lightweight structures he argued that, in order to make a structure light, what is needed is not a large number of light members with short lengths, which results in a commensurate number of connections, but an appreciably smaller number of heavy members. In *Structures Implicit and Explicit* he wrote:

> Look how wrong thinking can be: it was a great mistake, which I realized a good many years later, to say that the art of building is to build with matchsticks, that if you want to build light structures you must use light members because if you use light members a group of light members will be light. It took me quite a long time to see that it was just the opposite; it's the art of making a light structure with big, heavy members. A paradox. But not when you take dimension into account.

Le Ricolais stressed the importance of hierarchy in a structure, citing the example of skeletons and other natural forms. Both Galileo and D'Arcy Thompson had spoken of the optimum ratio between weight and size in relation to bone structures and skeletons which imposes a size-limit on the animal frame. Le Ricolais was always searching for paradoxes, which he applied to his design experiments. His enduring goal was to achieve the most famous of the paradoxes that he had put forward: a structure of zero weight and infinite span. His admiration for nature is summarized in another paradox he identified, that between strength and fragility in the structure of an eggshell – a phenomenon Frei Otto had also observed in the silk threads of spiderwebs. According to Otto, the BIC value, or measure of constructive efficiency, of spiderweb silk surpasses that of steel.

For André Malraux Le Ricolais was the father of space-frames, though, characteristically, Le Ricolais maintained that the space-frame was not his own invention but had existed in nature for 300 million years. The same applied to Fuller's geodesic domes, which according to Le Ricolais had been predated by 300 million years of



STIFF HOLLOW ROPE

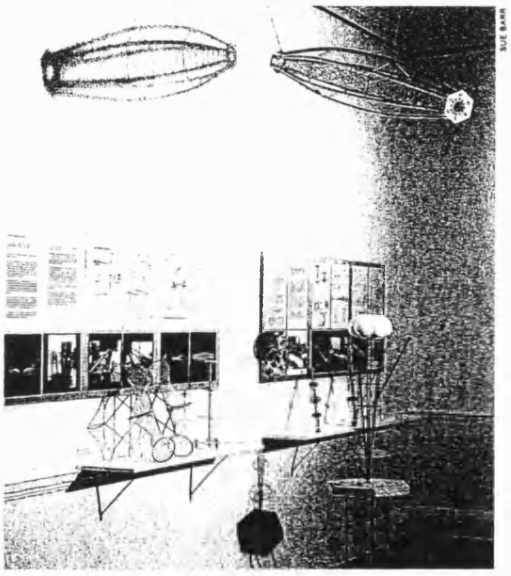
Left: *Euplectella*, glass sponge, from: R. E. Barnes, *Invertebrate Zoology* (1974). Compression cores, membrane skins. Centre: Micrograph of a bone. The art of structure is where to put the holes. Right: Testing Le Ricolais's tension-net bridge for the Skyrail of 1962–3. Interlaced and twisted steel cable around hoops form a stiff hollow structure.

radiolarian skeletons. His central point about nature was that it is not repetitive. He stressed the irregular curvature of our world, as observed by Leonardo in his studies of ballistic curves, and mathematically described by Newton with calculus.

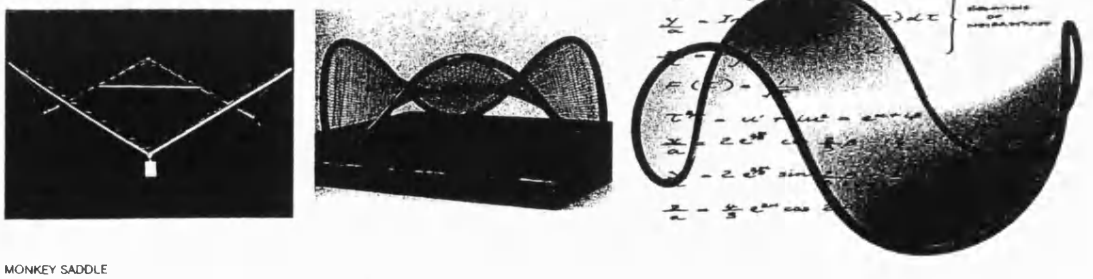
A great deal of Le Ricolais's inspiration came from mathematics. For him it was a form of elegant symbolism which performed the heroic task of simplifying. He equated mathematics with poetry and magic, though he acknowledged that physical models are ultimately more explicit and legible: it is the need for representation that converts intangible, abstract symbols and numbers into objects that breathe life. Indeed he insisted that he was more interested in the philosophical aspect of mathematics – the part that emphasizes relationships rather than its capacity for precision. In 'Introduction to the Notion of Form' he wrote:

> Before discussing form, it is of some help to become familiar with Gauss's concept of space. As early as 1816 Gauss had come to the conclusion that the famous Euclidean postulate about two lines meeting at infinity was impossible to demonstrate, and that the time had come for a new geometry in which there could be more than one parallel to a straight line passing through a point. Here we can witness the complete destruction of the Kantian principal of spatial intuition. This was a turning point in mathematical history, showing that reality and mathematics have nothing in common, confirming the saying of Renan: 'everything is fruitful save common sense'.

It is not clear from such passages whether he took the trouble to study this mathematics in depth or whether he was guilty of the charge that Sokal and Briémont levelled at other French intellectuals such as Lacan, Irigaray and Baudrillard, that of using mathematics which they do not understand to make an apparently clever philosophical point. The claim that reality and mathematics have nothing in common is false, since it is work like Gauss's that led to Einstein's general theory of relativity, firmly rooted in reality, in which stress becomes a



View of the installation in the AA Exhibition Gallery.



MONKEY SADDLE

Left: Le Ricolais's study model of the Monkey Saddle, a warped hexagonal frame with stressed cables defining a minimal surface. Centre: Final model of the Monkey Saddle. Right: The Monkey Saddle revisited. Computer model by the authors, derived from the equations of Weierstrass.

purely geometric quantity – the space-like part of a tensor derived from the Ricci tensor. Perhaps it does not matter if the mathematics is not fully understood. A logical and consistent philosophy does not necessarily result in good architecture, just as good architecture can come from a non-sensical philosophy.

In the AA exhibition, there was a model of the 'monkey saddle' – a surface with three high points and three low points, one for each leg and the tail. This inspired us to revisit Le Ricolais's idea by producing a mathematical model of his monkey saddle (see illustration p. 59, bottom right). Our version consists of a soap-film, or minimal surface, produced by using the equations of Weierstrass, in which a minimal surface is associated with an analytic function of a complex variable.

The exhibition of Le Ricolais's work served as a reminder of connections that exist between mathematics, nature and engineering structures – interests which we have been pursuing in our own work and which many architects and engineers have been keen to explore. We will revisit Le Ricolais again and again.

Bibliography

Bryan, James, and Rolf Sauer, 'Interviews with Robert Le Ricolais: Things Themselves Are Lying, and So Are Their Images', in *Structures Implicit and Explicit*, *VIA*, vol. 2, 1973.

David Georges Emmerich, Architecte-Ingénieur: Une utopie rationnelle, Collection of the FRAC Centre, Musée des Beaux-Arts d'Orléans, 1997.

Emmerich, David Georges, 'Histoire de l'A.R.S. – Le Ricolais', *Dossier de l'Atelier de Recherche Structurale UP6* (Paris, 1983), p. 190.

Hill, Anthony (editor), *DATA: Directions in Art, Theory and Aesthetics* (Greenwich, Connecticut, 1968).

Le Ricolais: Espace, Mouvement et Structures, Musée des Beaux-Arts, Nantes, 1968.

Le Ricolais, Robert, 'A la recherche d'une mécanique des formes', conference paper at the Palais de la Découverte, July 1965, in conjunction with the exhibition *Le Ricolais: Espace, Mouvement et Structures*, reproduced in R. Motro, 'Formes et Forces dans les systèmes constructifs, Application au cas des systèmes réticulés, spatiaux, autocontraintes' (thesis, Académie de Montpellier, Université des Sciences et Techniques du Languedoc, 1983).

Le Ricolais, Robert, J. S. Okie and C. Sawyer, 'Diamond Network Systems', in *Proceedings of the Second International Conference on Space Structures, Department of Civil Engineering*, University of Surrey, Guildford, September 1975.

Les Pratiques de l'espace (Paris, 1983).

McCleary, P., 'Robert Le Ricolais', *Lotus International* 99.

Mimram, Marc, *Structures et Formes: Étude appliqué a l'œuvre de Robert Le Ricolais* (Paris, 1978).

Robert Le Ricolais, Visiones y Paradojas (Madrid, 1997).

Top: Spiral sculpture. Above: Sinusoidal bridge study. Computer models by the authors.

Frei Otto

Modern research investigating the use of natural forms in relation to architecture and engineering has been pioneered by the architect, Professor Frei Otto, and his colleagues in Stuttgart, in collaboration with the engineer, Professor Sir Edmund Happold, with his colleagues in Bath. This collaboration began over thirty years ago when Ted Happold was leading Structures 3 at Ove Arup and Partners. At the time Peter Rice was working for Ted Happold and their particular interest was in the form generation, design and structural analysis of lightweight, long-span 'unconventional structures.' Both Ted Happold and Peter Rice have recently died.

The IL[§] has published Frei Otto's collaborative research work, where frequent reference is made to biological structures, both plants and animals. The Institute's publications record a series of experiments that set out to grasp rules and techniques that natural organisms employ in self-formation. These rules were analysed by Otto and his team with a view to providing a method for generating form in architecture.

Biological structures were thought to be optimum and this belief reached a quasi-religious status, particularly in the 1960's and 1970's. Much research was done by architects, engineers and biologists and presented at conferences in Stuttgart and in IL publications. It is probably true to say that little of this work was truly interdisciplinary in that most publications were architectural, engineering or biological. Much of this research work was accessible only at seminars and conferences on the evolution of natural structures organised by the SFB 64[‡] and Structural Morphology groups such as the IASS[¶].

[§] *Institüt für leichte Flachentragwerk,* a research institute of the University of Stuttgart set up by Frei Otto in the 1960's carrying out inter-disciplinary work in the fields of Biology and Building.

[‡] *SonderforschungsBereich 64,* an off-shoot of the IL and Collaborative Research Centre established at the Universities of Stuttgart and Tübingen since 1984 under the sponsorship of *Deutsche Forschungsgemeinschaft* setting out to gain full understanding of structures in technology and biology.

[¶] International Association for Shell and Spatial Structures founded in 1959 by Eduardo Torroja *et al.* to facilitate innovative design for shell and space structures.

Otto's most recent work produced in collaboration with Rasch (1996) is a synthesis of past and present research. It presents for the first time a concise overview of theory and practise relating to Otto's involvement in projects over the past thirty years, which are expressive of his organicist design philosophy.

One aspect of Otto's work is of particular reference to this study; he repeatedly put forward ideas for architectural forms in his published research. Thus he felt that it was acceptable for an architect to use architectural skills and creativity in research, rather than become an historian, philosopher or specialist in some technical discipline. Some architects believe that Otto was more like an engineer, but engineers who worked with him are in no doubt that his thought process was much more that of an architect.

Heinrich Hertel's studies were being conducted at about the same time as Otto's research although they dealt with the broader scientific implications of natural analogy in relationship to engineering and aeronautical science. Figures 1.6.1a and 1.6.1b show examples of his work.

1.6 Other research workers

1.6.1 Frazer, Vincent

Frazer

Other work of relevance to this study is research work by John Frazer and his students. Frazer (1995) in his book entitled *An Evolutionary Architecture* describes research carried out at the Architectural Association spanning thirty years. The research was concerned with the development of a self-evolutive design experiment that was modelled on organic process. Using automotive techniques that were similar to nature's genetic instructions, an interactive model was bred from a seed, and then cultured to adopt cell propagation behaviour driven by man-made coded instructions and changing environments.

It is perhaps the aims, philosophy and methodology of Frazer's work at the AA that are closest to the aspirations of this research. Frazer discusses the implications of attributing design capabilities to nature. He acknowledges the questions raised about Darwin and Dawkins on the chaotic force of natural selection on the one hand, and the

opposing theories of Paley and Pettigrew on the purposiveness of a first cause and a *designer* on the other.

Vincent

Biomimetics, set up by Julian Vincent in the mid-seventies at Reading University further explores the potential for a scientific application of nature's materials, method and process. Vincent defines Biomimetics as,

'the abstraction of good design from nature. It is the process by which ideas from biology can be applied within such disciplines as chemistry, engineering and materials science. Biomimetics is a multi-disciplinary science where ideas from nature are harnessed by Biologists, Material Scientists, Chemists and Engineers and used to design new "smart" materials or structures to perform specific functions'[§]

For Vincent and his colleagues, nature's techniques and materials form a continual basis for the development of innovative techniques, which create new possibilities in material science.

An example of a biomimetic device is illustrated in figure 1.6.1a, showing the microstructure of a plant bur, which forms the basis of the proprietary Velcro hook and loop fastenings shown in figure 1.6.1b.

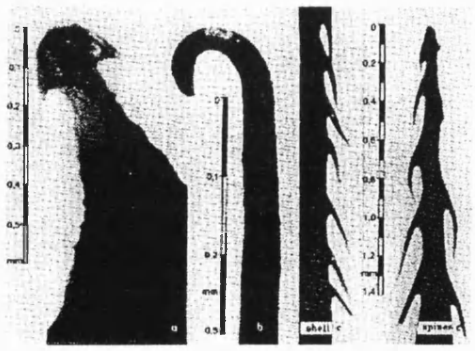


Figure 1.6.1a Bur spines with barbs. *Hertel (1963)*

[§] Vincent, J.F.V, *Borrowing the Best from Nature* and the information publication of the Centre for Biomimetics, University of Reading, UK, undated.

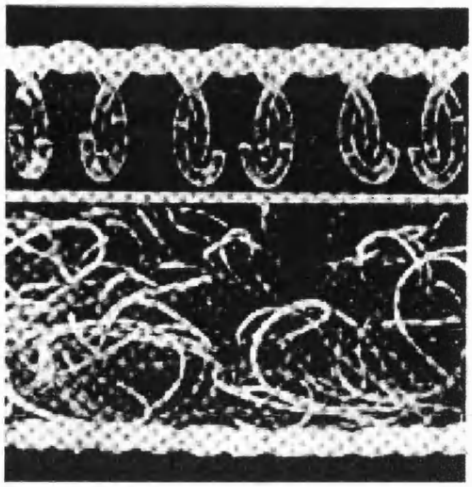


Figure 1.6.1b Hook and Loop fastening. Hertel (1963)

Biomechanics is a scientific application of nature's method and process. It attempts to repair tissue and restore joint and motor function in circumstances where irreversible damage to the body's locomotive parts has taken place by *copying* the techniques nature uses.

It is clear that nature continues to inspire artists, scientists, architects and engineers and despite the apparent disparity between these disciplines, they all remain connected by the universal call of nature.

Chapter 2.0 The Anatomy of Natural Form

Before introducing the visual material in this chapter, it may help to define the broad limits associated with the term natural form. Lissitzky (1924) describes the term *nature*, from which the term natural form is derived, as originating from the Latin, *nasci*, which means to become or to develop, and therefore concludes that nature *'is everything that develops, moves and forms itself out of itself through its own force.'* The results of these involuntary processes are evident in both organic and inorganic formations.

Steadman (1979) does not use the term *natural form*, neither does he favour the use of the term *organic form*; he explains that it has too wide a connotation. In this investigation, the use of the term, *natural form*, is deliberate since it allows the author to highlight the common form characteristics found in organic and inorganic matter, rather than focus on their obvious differences. Pettigrew (1908) suggests that in fact, the persistence of certain characteristics across the entire board of non-living and living organisms serves to blur the boundaries between the living and non-living world, reinforcing the notion of a Universal Law of Nature and standing as proof of a First Cause. Stevens (1974) supports this view by referring to the immense variety that nature creates by *'the working and reworking of only a few formal themes.'* For him the limitations produce the variety.

This chapter presents a series of images that have been chosen to highlight the aesthetic and dynamic qualities of natural forms. It recalls themes in the work of architects discussed in the previous chapter and attempts to focus on features that provide opportunities for design.

The images have been selected from the proliferous anatomical studies by Pettigrew (1908), the vivid and larger than life photographic records of nature by Feininger (1966), a vast collection of photographs depicting rare shells by Stix (1969), the analytical diagrams by Hertel (1963) relating the mechanics of animal locomotion to aeronautics and naval engineering, and various photographic material from publications of the Institute for Lightweight Structures (the IL).

Pettigrew's three-volume text on *Design in Nature* uses the multiplicity and complexity of forms in nature as evidence of the work of a creative intelligence, or God. Paley (1802), before Pettigrew, had also constructed this argument in his '*Natural Theology - or evidences of the existence and attributes of the Deity collected from the appearances of nature.*' He believed that such forms could not have evolved through Darwinian notions of survival of the fittest. More recently Dawkin (1986) has presented a neo-Darwinian standpoint on how evolution could have produced such complexity.

The framework of structure, curvature and movement provides a basis for analysing persistent themes in nature. Some of these themes have been highlighted in Chapter 1 in discussing, for example, Horta's moving line and also, the work of the great shell engineers, Nervi, Torroja and Candela.

There is further close correlation between the arrangement of skeletal elements in radiolarian shells and Le Ricolais's triaxial networks. Gaudí used bone forms to enliven the structural elements of his buildings. Pier Luigi Nervi, Felix Candela and Eduardo Torroja consistently applied a sculptural approach to structural form reminiscent of shells and of the skeletal structures of bones and trees. Calatrava invites dynamism into his designs for bridges and pavilions, through the use of curvilinear geometry.

2.1 Structure (as ordered parts)

In a general design sense, the term *structure* has two meanings. On the one hand it refers to *ordering systems,* for example, those that exist in the lower forms of life and in the minute parts of plants and animals. These minute parts, structured of smaller parts themselves, come together in a careful arrangement to form the components of higher ordered systems, which ultimately form a cohesive whole. Radial symmetry, radial segmentation and concentric ring formations are predominant structuring devices used by nature at both the micro and macro level.

The other meaning of *structure*, self-support is discussed in section 2.2.

2.1.1 Shape Homology, Differentiation, Cleavage

Shape Homology which refers to a correspondence of form between organisms with different functions, *differentiation*, the ability to assign varying form characteristics to

appendages or limbs performing similar functions and *cleavage*, the tendency for splitting to occur, are further structuring devices of nature.

Figures 2.1.1a – 2.1.1l show examples of homological shape arrangements in organic and inorganic structures, demonstrating Pettigrew's conviction of universal law.

Figure 2.1.1a shows a magnified drawing of crystal of calcium carbonate, showing almost identical radial symmetry and markings to the transverse section of oak in figure 2.1.1b. Figure 2.1.1c shows an arrangement of concentric rays with a centre locus in the vertebra of shark, which is comparable to figure 2.1.1d showing a display of iron filings held by magnetic force.

The fungus in figure 2.1.1e shows irregular ring formation and a ragged perimeter that can be compared with the shell of a limpet in figure 2.1.1f. Apart from their moist rocky habitats, they share no common ancestry.

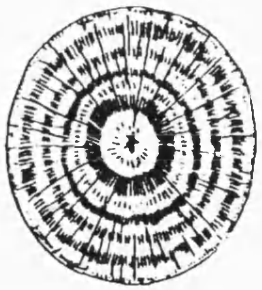


Figure 2.1.1a Calcium carbonate crystal

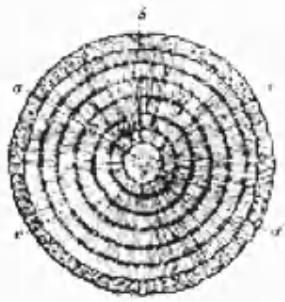


Figure 2.1.1b Transverse section of oak stem

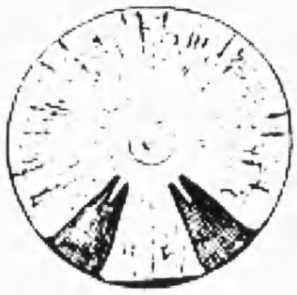


Figure 2.1.1c Vertebra of shark (section)

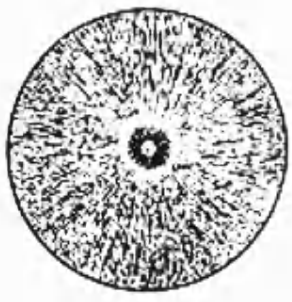
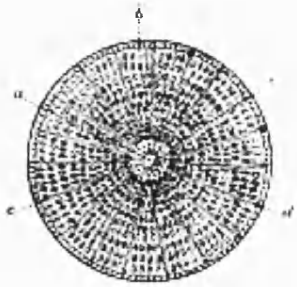


Figure 2.1.1d Iron filings held by magnetic force

Figure 2.1.1e Fungus (*Hexagonia glabra)* **Figure 2.1.1f** Limpet (*Fissurella nimbosa)*

In figures 2.1.1g and 2.1.1h, the transverse section through a diatom shell and section through a sea urchin show segmentation of their cellular structures and a centre core. Figures 2.1.1i and 2.1.1j show radial ribbing of deep-sea coral and crystal of strontianite, a component of minerals.



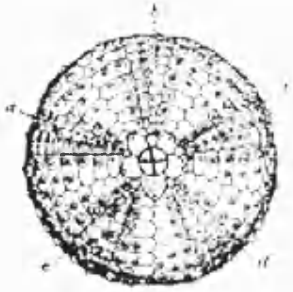
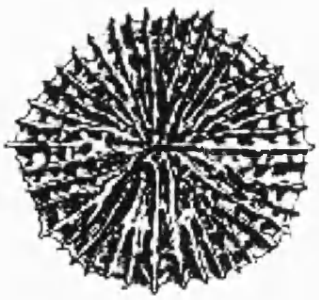


Figure 2.1.1g Section of Diatom

Figure 2.1.1h Urchin section



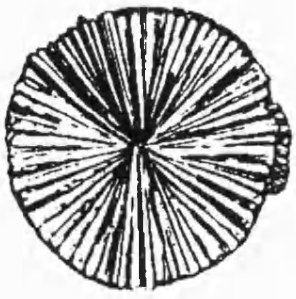


Figure 2.1.1i Deep-sea Coral

Figure 2.1.1j Crystal of strontianite

The concentric accretion rings in the vertebra of a shark in figure 2.1.1k resemble those of the section of a touch corpuscle in figure 2.1.1l, although the corpuscle has been greatly magnified.

Figure 2.1.1k Vertebra of shark (section)

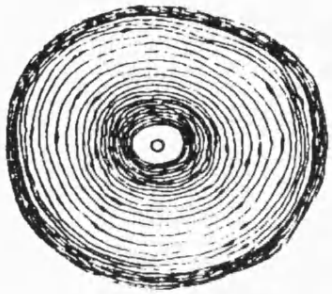


Figure 2.1.1l Touch corpuscle

These themes of radial symmetry, concentricity and radial segmentation have for centuries provided the basis for ordering systems in ancient and Classical architecture, and still feature heavily in space organisation in modern design. Some of these geometrical themes are evident in the conformal maps from which the design studies in chapter 4 have emerged.

2.2 Structure (as self-support)

Structure, on the other hand refers also to the self-supporting frameworks that exist in nature, enabling organisms to maintain their shape characteristics during growth and movement, or when subjected to the forces of the external environment.

2.2.1 Stiffening, Branching, Membrane networks

Stiffening by bracing, strutting and ribbing are common stabilising devices that occur in bone structures, plant stems and wing membranes. Folded plate action and compound curvature are further structural devices used in mollusc shells, plant bodies and wing structures. Figures 2.2a – 2.2i illustrate these stiffening mechanisms.

Figure 2.2a shows folded plate stiffening of the cock's comb oyster while figure 2.2b shows the fan-like arrangement of stiffening ribs in the costate cockle, which recall

Nervi's great halls. Figure 2.2c shows rather unusual stiffening by rounded fins in the Lamarck clam.

Figure 2.2a Cock's comb Oyster (*Lopha cristagalli*) Stiffening by folded plate

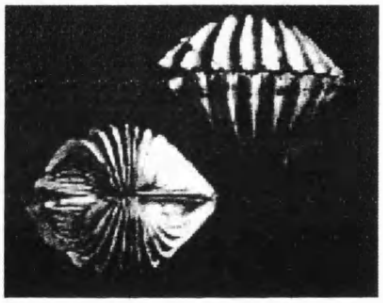


Figure 2.2b Costate cockle (*Cardium costatum Linné*) Stiffening by ribbing

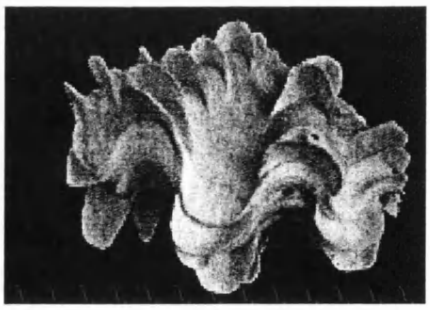


Figure 2.2c Lamarck clam (*Tridacna squamosa*) Stiffening by rounded fins

In figures 2.2d - 2.2f, the shell of the king crab, the wing bone of the eagle and skull capsule of the Tawny owl all show hollow casings with almost identical vertical strutting,

despite the variation in their function and position in the animal body. In the case of the Tawny owl skull, the stiffening is multi-storeyed.

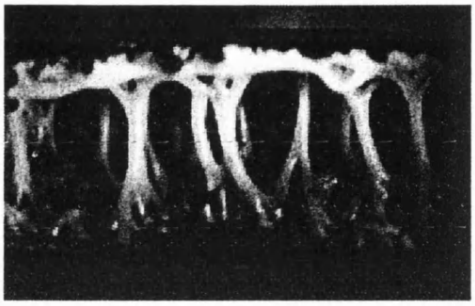


Figure 2.2d Shell of King Crab, section. *Feininger(1956)*

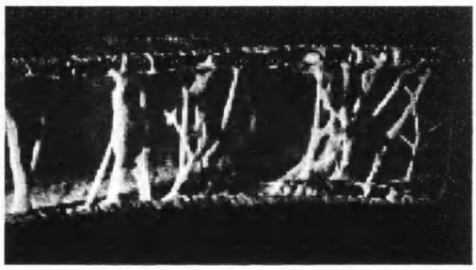


Figure 2.2e Wing bone of Eagle, section. *Feininger (1956)* Stiffening by struts and braces

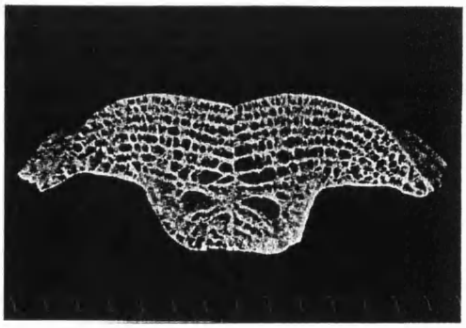


Figure 2.2f Skull capsule of Tawny Owl, section. *IL6 (1973)*Multi-storey stiffening

Further skeletal stiffening is shown in the bone structures in figures 2.2g and 2.2h.

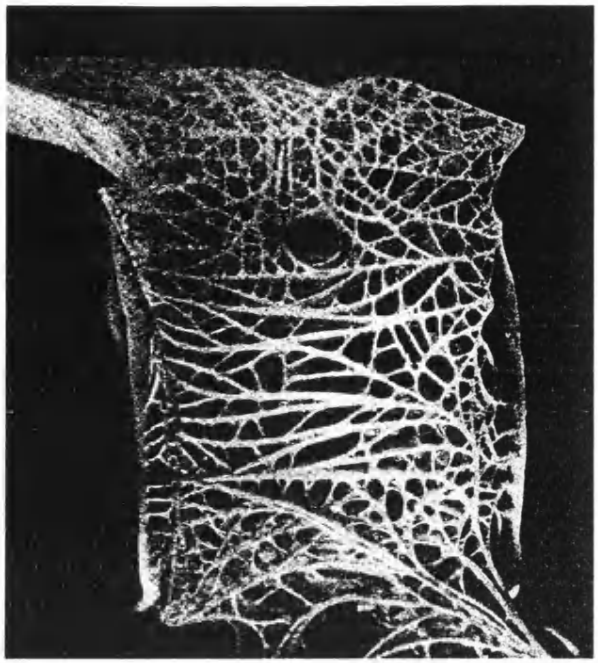


Figure 2.2g Section of vertebra bone. *Feininger (1966)*

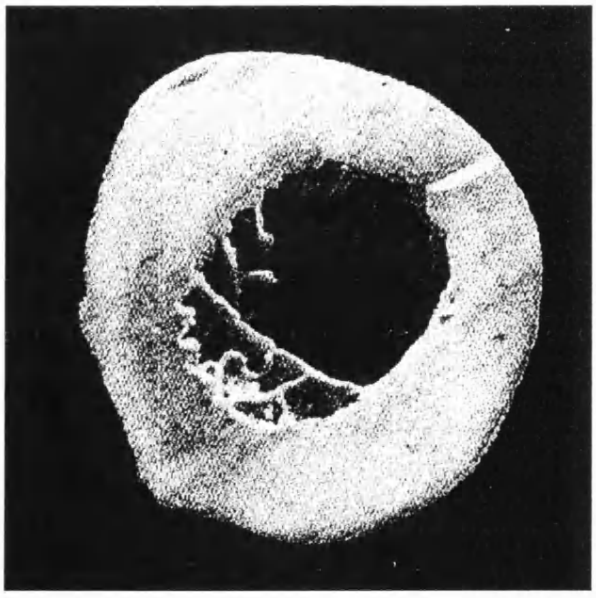


Figure 2.2h Cross-section of humerus. *Feininger (1966)*

The wing membranes in figures 2.2i and 2.2j show stiffening combined with a thin connective membrane, which allows the wing panel to yield under wind pressure.

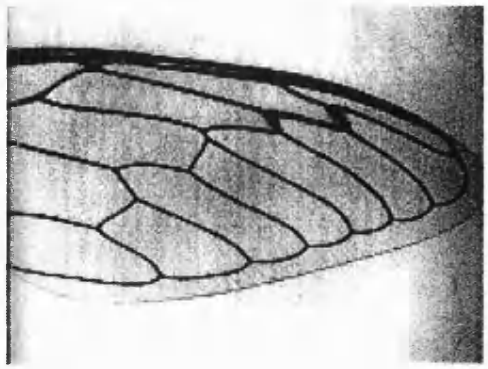


Figure 2.2i Wing membrane. *Feininger (1956)*

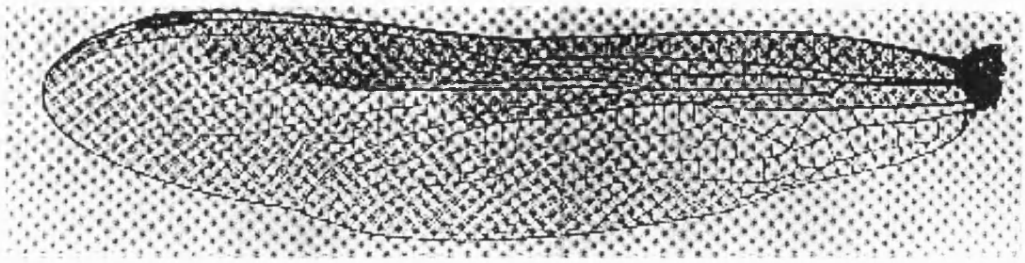


Figure 2.2j Blue dragonfly wing. *Hertel (1963)*

Similar stiffened networks exist in the leaves of plants. In particular the veined structure of the *victoria regia* water lily with a span of 2m is able to be loaded without collapsing or sinking. Paxton, the gardener, was inspired by this model to create glass and steel greenhouses, notably the Crystal Palace of 1851.

Le Ricolais developed stiff, flexible structures based on his dictum, *stiff hollow rope.* His structures recall the intricate lattice structure of the glass sponge, *Euplectella.*

Relatively floppy structures such as animal intestines and blood vessels maintain their form to some degree by using the pressure of the fluid or air passing through them, or by their attachment to a framework. These structures inspired Frei Otto in the design and development of inflatable structures.

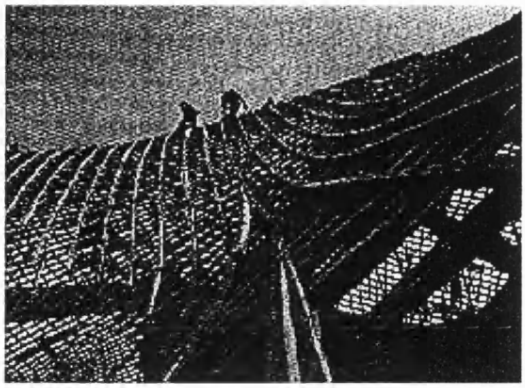


Figure 2.2k The Mannheim Bundesgartenschau under construction

The *Bundesgartenschau* grid shell in Mannheim shown in figure 2.2k is created from a two-layer system of 5cm x 5cm timber laths, pinned at joints with bolts and spring washers. Its final stiff state was achieved by the lifting of its initially flat grid into shape using spreaders on scaffold towers.

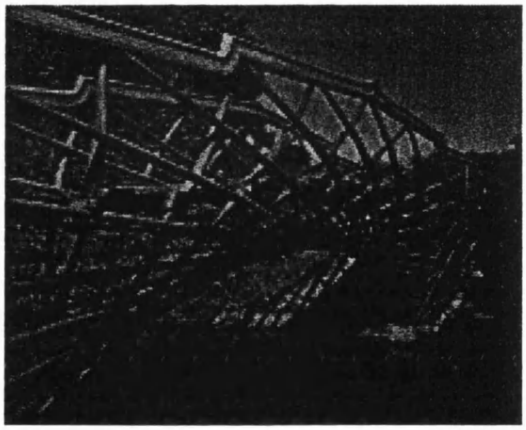


Figure 2.2l Pedestrian bridge for Eton College by Jamie McCulloch

Figure 2.2l shows a curved lattice pedestrian bridge for Eton College designed and built by Jamie McCulloch. Its stiff structure is achieved by the use of a 3-dimensionally curved timber framework.

2.3 Curvature (as points in continuous motion)

Curvature is a powerful theme in which the forms of nature are frequently expressed. As well as being a feature of shells, spiralling forms are found in the many internal

arrangements of animals, in the external arrangements of plant stems and in the horns and bones of animals.

Cook (1903) emphasises the inextricable link between curvature and natural form. Pettigrew (1908) insists that curvature and the tendency to spiral result from life force, growth and rhythmical change. D'Arcy Thompson discusses spiral phenomena at length and their amenability to mathematical analysis. Curvature is not only a dominant characteristic of the *forms* of the natural world; it is also an expression of movement and growth.

Von Seggern (1993) regards curves '*as being abstractions of the form and motion of the physical world. Scientists have analysed this world for millennia in order to render these abstract expressions in the most minute detail, from gross astronomical movements to infinitesimal atomic phenomena.*'

The fact that many living organisms assume curved forms or grow and move along curved paths is not incidental. Curvature is an essential function of the mechanism of smooth movement and growth of natural forms.

2.3.1 Inorganic and Organic Curvature, Spiral Phyllotaxis

Figures 2.3.1a – 2.3.1f show curvature and spiralling both at the microscopic level and in structures visible to the human eye. The human spine in figure 2.3.1a and the horn of addax in figure 2.3.1b both show undulating curvature and signs of cleavage and annulation. Tight spiral formations are typical of the fern illustrated in figure 2.3.1d, and also appear in the crystal of sulphur in figure 2.3.1c and in the crystal of prochlorite in figure 2.3.1e.

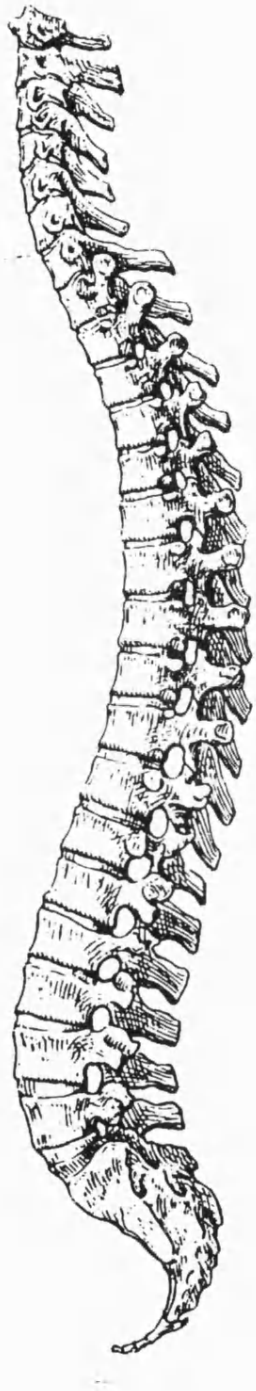


Figure 2.3.1a Human backbone

Figure 2.3.1b Horn of Addax

Curvature, Cleavage, Annulation and Spiralling

Figure 2.3.1f shows an example of counter-clock-wise spiral *phyllotaxis* in the section of pinus pinea or pinecone obeying a 5 + 8 or 8/13 Fibonacci fraction.

Figure 2.3.1c Crystal of Sulphur **Spiralling due to rapid cooling**

Figure 2.3.1d Fern frond **Spiralling in growth**

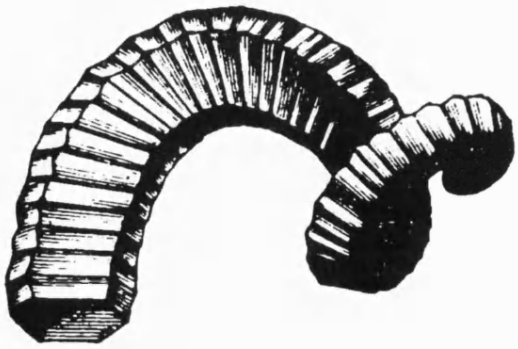


Figure 2.3.1e Crystal of Sulphur **Spiralling and segmentation**

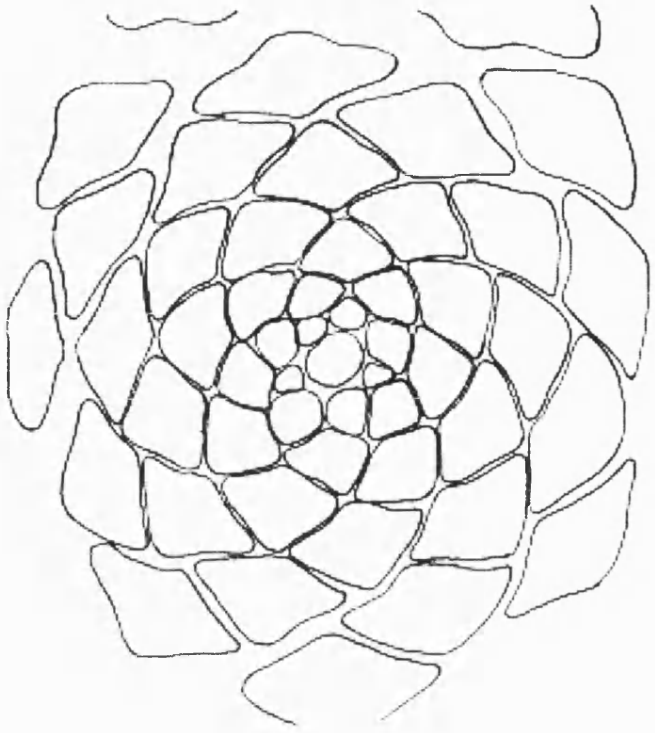


Figure 2.3.1f Section of pinecone (*pinus pinea*) sketched after Cook (1979)

Spiral Phyllotaxis

2.4 Movement (as curved lines)

D'Arcy Thompson and Holton (1965) speak of the scholastic adage, '*ignorato motu ignoratu natura'*, translated as *'who knows not motion knows not nature.'*

In the context of this dissertation, movement not only refers to locomotion in animals, but also to the effects that wind and organic erosion have on ground planes and their inhabitants.

Feininger gives many examples in which inorganic movement has left its trace on a site. Figures 2.4a, b and c show evidence of this movement. Imprints or markings on rock faces and tree barks, or indents traced into the sand show the natural beauty of organic erosion These markings inspire artists because of their organic quality.

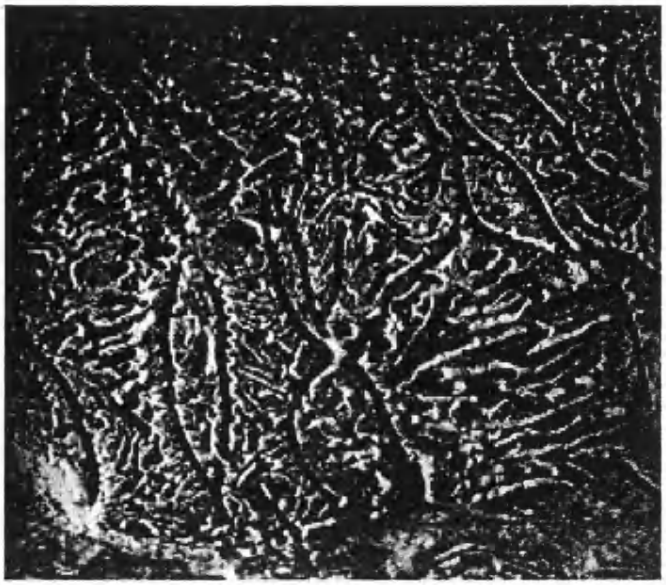


Figure 2.4a Curved tunnels carved by the larvae of bark beetles

Figure 2.4b Nests in heart wood carved by carpenter ants

Figure 2.4c Erosion of Bryce Canyon, Utah

2.4.1 Inorganic, Organic and Celestial

Curvature is closely linked with movement, because the path marked out by objects, in moving from one position to another, or in passing from one stage of growth to another, is often curved. In fish swimming and in birds flying, these soft bodies use sinusoidal trajectories to overcome the great forces of air and water resistance whilst trying to minimise effort.

Curvature is an aesthetic and dynamic feature of natural forms; it appears to suggest movement. Movement is an essential characteristic of living nature; it ensures survival. Morphogenetic movement, of which cell division is an example, means that nature's forms are continually being transformed or renewed.

Planetary motion is an example of a higher order of curved movement. The paths of celestial bodies are curved. It was this characteristic which helped Einstein to solve the mystery of gravity by applying theories of classical differential geometry, (the geometry of curved lines and surfaces) to planetary motion in producing his General Theory of Relativity in which the planets are not acted upon by a force, but move along geodesics in space-time.

We are structured of curved parts, our movement and growth are governed by curvilinear characteristics, the world we live on is a spherical body, this spherical body rotates continually around a polar axis; the planets, the moon, the stars and the sun move around us along the ecliptic in an elliptical path. Curved laws not only govern our forms, but also the environment in which we live.

Figures 2.4.1a – 2.4.1c show the curved trajectories of swimmers and flyers. Figure 2.4.1a shows the diminishing wave-like lashing movement of the bull's sperm in forward motion.

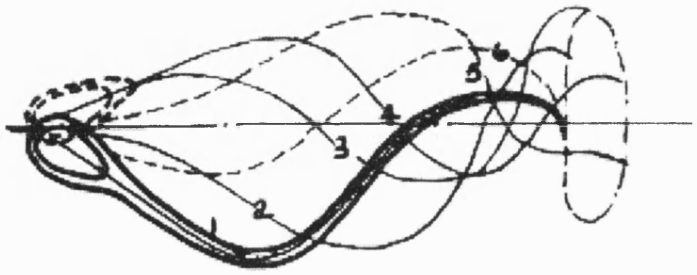


Figure 2.4.1a Forward stroke of sperm of bull sketched after Hertel (1963)

A bird is illustrated in figure 2.4.1b, showing the sinusoidal flight path plotted by its wrist joint in the down and upstroke movement of the wing.

Under the effect of a sudden electrical impulse, the fast start of a trout is recorded in figure 2.4.1c. The trout itself appears rigid in the stationary position, but being a *soft body,* yields to form a gentle curve, in its attempt to develop the *thrust* to overcome the resistance of the water medium in which it swims and lives.

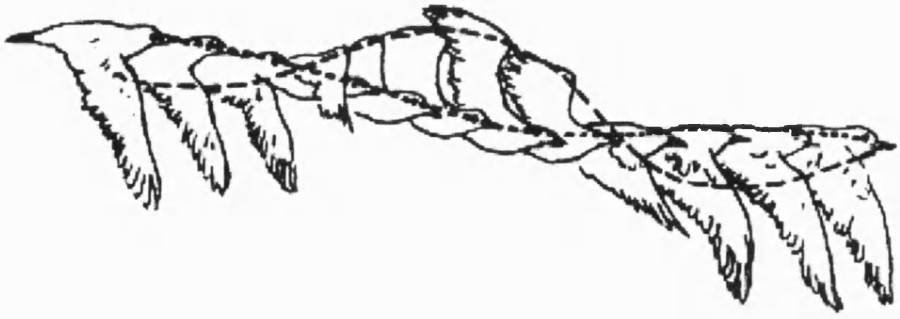


Figure 2.4.1b Flight path at wrist joint of bird sketched after Hertel (1963)

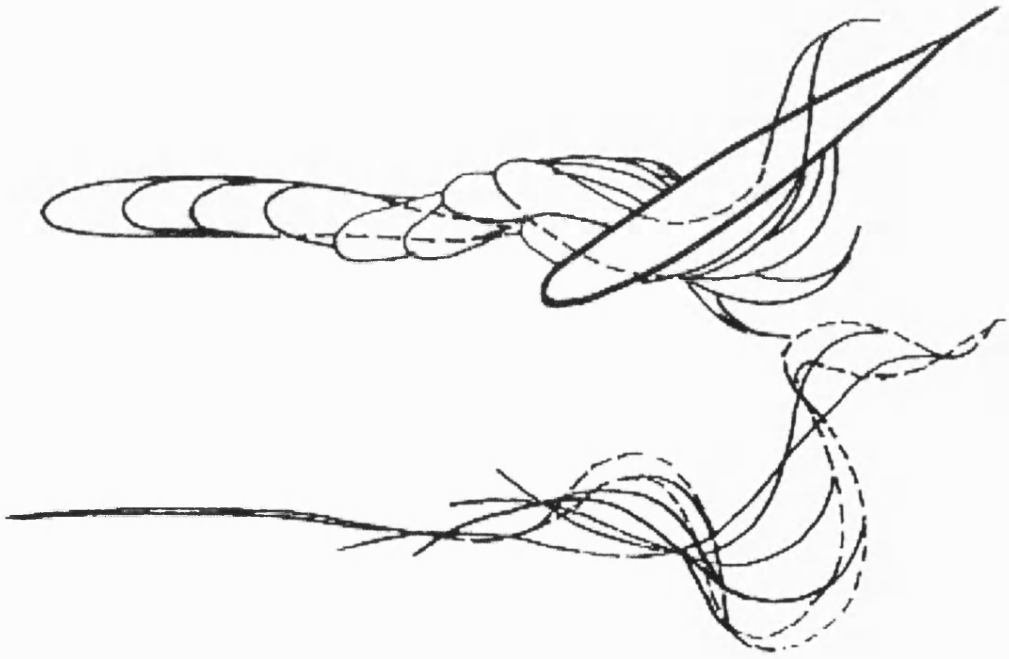


Figure 2.4.1c Fast start of Trout sketched after Hertel (1963)

Sinusoidal movement paths of swimmers and flyers

Having constructed the building blocks that nature uses in design, the next chapter presents mathematical concepts that will assist in developing methods to generate curvilinear form for application to architecture.

Chapter 3.0 Mathematics, Computers and Curved Form

Before introducing the design studies in the next chapter, the present chapter seeks to give a brief outline of the historical development of mathematical theory with particular emphasis on curvilinear geometry. The invention of the differential calculus by Sir Isaac Newton and Leibniz, is considered to be the watershed in the history of mathematical analysis, paving the way for subsequent developments in more complex techniques of describing curved form.

After giving a brief outline of the development of curved geometry, the concept of topology is introduced, which has many general applications in modern mathematical analysis. Following this, the discussion will be centred on methods of describing curved lines and surfaces, spiral geometry and complex number theory.

These methods are supplemented in Appendix A1.0, A2.0 and A3.0, with exercises that combine the mathematical principles with simple computer programs to illustrate their practical application. The exercises are to prepare for the more advanced techniques of geometry finding that will be explored in the design studies.

3.1 Curvature and Movement in Mathematics

3.1.1 Newton, Einstein

Newton (1642-1727)

Setting aside for a momemt the contributions of classical geometers, Sir Isaac Newton emerges as the central figure in the development of the mathematical description of curved geometry. Some time between 1665 and 1666, he discovered the method of differential and integral calculus. Differential calculus is described in more detail in section 3.1.2. Newton's theories of calculus and his preoccupation with the search for answers to questions about the form, process and movement of the universe culminated in 1687 in the publication of his grand text, *Philosophiae Naturalis Principia Mathematica* (The Mathematical Principles of Natural Philosophy). His later texts, *De quadratura curvarum* (On the quadrature of curves published as an appendix to his *Optiks* in 1704), *Analysis per quantitatum series* (Analysis by Means of Various Series, 1711) and the

posthumously published, *Methodis fluxionis* (The Method of Fluxions, 1736), gave fuller details of his methods of calculus.

Gottfried Wilhelm Leibniz (1646 - 1716), whose independent discoveries relating to calculus had occurred in 1676 were published as, *Nova methodus pro maximis et minimis* (A New Method for determining Maxima and Minima), in 1684, before Newton's. This led to some confusion over the proper authorship of the method of calculus. It is accepted now that the theory of infinitesimal calculus can be attributed to both Newton and Leibniz. Leibniz has also been credited with the earliest attempts to replace the traditional logic of Aristotle with mathematical logic.

Clearly, Newton realised that the key to an accurate description of the universe lay in the formulation of a theory of curved geometry. For him and many others, curvature was a crucial characteristic of the universe and its elements. It not only described the *form* of the universe and its elements but also the *forces* that these elements were subject to and the *motions* that arose in an effort to overcome these forces.

Ballistic curves which had been sketched accurately by Leonardo in *Codex Madrid I* but which he was unable to calculate, were solved mathematically by Newton in 1687. Galileo, had also applied himself to the problem of predicting the curved path of a projectile moving through the air, but had neglected the effect of air resistance and so had produced less accurate curves.

Newton's discovery of calculus was followed by the contribution of Jakob, Johann and Daniel Bernoulli whose knowledge of infinitesimal calculus lead to the development of the calculus of variations. Jakob's work in particular was centred on applying calculus to the study of curves, especially the logarithmic spiral. Daniel, Johann's son is noted for his work on hydrodynamics, particularly his book *Hydrodynamica* published in 1738.

Euler, Lagrange (1736 - 1813) and Gauss (1777 - 1855) added to mathematical theories of curved geometry through progressive developments in theories of calculus.

Gauss in particular discovered certain rules that govern the nature of curved surfaces. The Gaussian curvature of a given point on a surface is defined to be equal to the product of the principle curvatures at that point. Positive Gaussian curvature is synclastic or convex, whilst negative Gaussian curvature is anticlastic or saddle-shaped.

Gauss discovered that the sum of the angles of a geodesic triangle on a surface is equal to 2π plus the surface integral of the Gaussian curvature within the triangle.

In 1823 and 1826 respectively, Bolyai the Hungarian, and Lobachevsky the Russian, independently discovered hyperbolic geometry. Hyperbolic geometry has particular relevance to Torroja's shell structures, many of whose directrices are hyperbolic paraboloids of revolution.

Einstein (1879 - 1955)

Einstein used the work of Gauss, Christoffel, Reimann, Ricci and Bianchi on the geometry of curved surfaces and manifolds to produce his General Theory of Relativity. In this theory the six components of the stress tensor in three dimensions become the ten components of the stress-momentum-energy tensor in four-dimensional space-time. The components of the stress-momentum-energy tensor depend upon the second derivatives of the metric tensor used to calculate the time interval between events. Thus stress, as well as mass, bends space-time and stress becomes a purely geometric entity. The Bianchi relationships in pure mathematics, give the equilibrium equation relating stress and momentum.

3.1.2 Topology, Differential Geometry, Curved Lines and Surfaces

Topology

The word topology comes from the Greek word, τοποζ, meaning 'a place.' Topology has become an important branch of mathematics because it studies the continuity or congruence of geometrical shapes, and an understanding of its principles has many important consequences in the field of modern geometrical analysis.

Topology is useful because it deals with the intrinsic relationships between elements rather than with the precise nature of their geometry. For example, a sphere, a tetrahedron and a cube are topologically equivalent to one another, although they differ in their geometry. They are topologically equivalent because any one of these figures can be transformed into the other by a process of continuous deformation without self-intersection. None of these figures is however, topologically equivalent to a torus. The hairy ball theorem illustrates one topological property of the sphere as distinct from the

torus. Although the entire surface of a torus can be covered with a bed of hair with all shafts of hairs lying flat and uni-directional, a sphere cannot; attempting to cover a sphere with a bed of hair always results in a hairless pole at the crown and a tuft elsewhere.

The principle of topology can further be illustrated by applying a pattern of lines to the surface of a rectangular lamina. Deforming the rectangle neither affects its topology, nor its surface pattern because the connections between all lines are preserved; its geometry, however, does change.

Topology has a particular relevance to D'Arcy Thompson's theory of conformal mapping because its principles enabled him to demonstrate interrelationships between specie despite obvious differences in morphology. Topology is a recurrent theme of nature. Through a series of continuous deformations, he showed how topological equivalence was maintained in certain fish, which indicated that these fish might possibly have evolved from other fish of seemingly different species. He suggested that morphological differences in themselves did not always indicate a difference of type.

Goethe also, in his essay on morphology of 1817 suggested topological classification systems when he formulated the idea of the primordial *Urpflanze*, or universal plant. He believed that a limited number of archetypal plants existed as models for all possible variations of the plant kingdom. One could suggest that Darwinian evolutionary theory is topologist in its argument.

Topology allows classifications to be made on the basis of inherent characteristics, rather than on outward appearances. This feature of topology makes it possible in modern mathematical analysis, to derive assumptions from simple figures, which can then be applied to other more complex shapes provided that topological equivalence is maintained.

Differential Geometry

Classical differential geometry is the branch of mathematics, which studies curved lines and surfaces using calculus. The word classical, as Williams points out, is used to distinguish it from the study of other curved objects such as that of four dimensional space-time to which the General Theory of Relativity relates. Williams recommends that the study of classical differential geometry is an ideal introduction into the mathematical methods needed for general relativity since the same notation can be used in both fields.

Both Struik (1950) and Faber (1983) offer good introductions to the subject of differential geometry for those with a good grounding in mathematics and additionally, Faber also deals with relativity theory.

Differential geometry studies the way in which points on curved lines and surfaces change in their orientation as one moves from point to point. To do this it uses the differential calculus, hence the name differential geometry.

Curved Lines

Let us consider a typical point A, on a curved line in three dimensions. The x, y and z co-ordinates of the point can be written,

$$x = x(u)$$

$$y = y(u)$$

$$z = z(u)$$

where u is a parameter. For example, a helix is given by

$$x = R\cos u$$

$$y = R\sin u$$

$$z = au$$

where R and a are constants. In this case the parameter is the plan angle the radius makes with the x axis as can be seen from figure 3.1.2a.

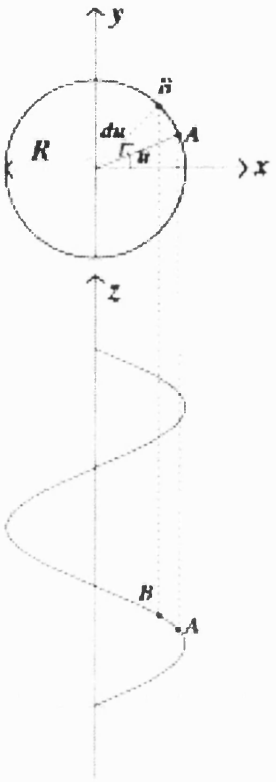


Figure 3.1.2a A typical helix.

Alternatively, it might be the arc length, which is usually given the symbol *s*. However, in general there is no simple physical interpretation of the parameter.

The point A can also be defined in terms of the position vector,

$$\mathbf{r} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$$

where **i**, **j** and **k** are unit vectors in the directions of the *x*, *y* and *z* axes. As *u* varies, *x*, *y* and *z* will vary and **r** will also vary. A second point B, has its parameter equal to $u + \delta u$ and the corresponding values of the co-ordinates are $x + \delta x$, $y + \delta y$ and $z + \delta z$. Its position vector will be,

$$\mathbf{r} + \delta\mathbf{r} = (x + \delta x)\mathbf{i} + (y + \delta y)\mathbf{j} + (z + \delta z)\mathbf{k}.$$

The ratio $\frac{\delta\mathbf{r}}{\delta u}$ is the change in **r** divided by the change in *u* between the two adjacent points A and B as shown in figure 3.1.2b.

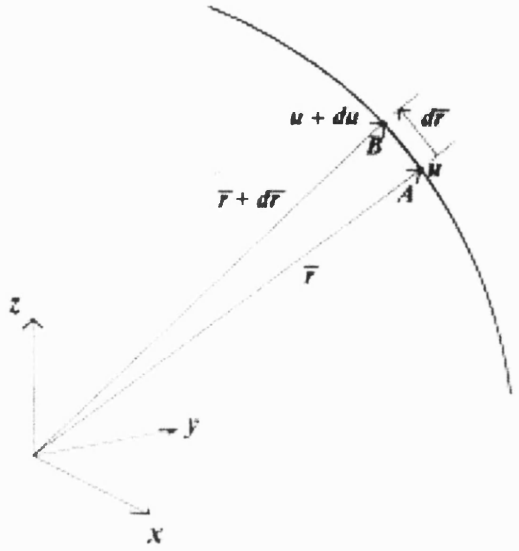


Figure 3.1.2b Diagram showing position vector, $\mathbf{r}$, with respect to the parameter, u

For a very small increment, or as δu tends to zero, we write that,

$$\frac{\delta \mathbf{r}}{\delta u} \rightarrow \frac{d\mathbf{r}}{du} \text{ as } \delta u \rightarrow 0, \text{ so that}$$

$$\frac{d\mathbf{r}}{du} = \frac{dx}{du}\mathbf{i} + \frac{dy}{du}\mathbf{j} + \frac{dz}{du}\mathbf{k}.$$

Curved Surfaces

In the case of a curved surface embedded in three-dimensional space, two quantities are required to specify a given location on the surface – equivalent to the lines of latitude and longitude on the Earth's surface. It is conventional to use a surface co-ordinate system, u and v, called curvilinear co-ordinates, which represent a grid of intersecting lines over the surface. Similar principles apply as with curved lines.

The Cartesian co-ordinates x, y and z are *mapped* to the surface co-ordinates so as to describe the particular shape properties of the surface in question. This is equivalent to saying that the Cartesian co-ordinates are functions of the two surface co-ordinates u and v,

$$x = x(u,v),$$

$$y = y(u,v)$$

and

$$z = z(u,v).$$

If u is kept constant at a particular value and v is varied, a series of x, y and z values will be generated which trace out a line on the surface. If this process is repeated for successive lines, each with a different value of u, which still remains constant along any one line, a family of lines is generated. A second family of lines can be generated such that on each line u is a constant and v varies as seen in figure 3.1.2c.

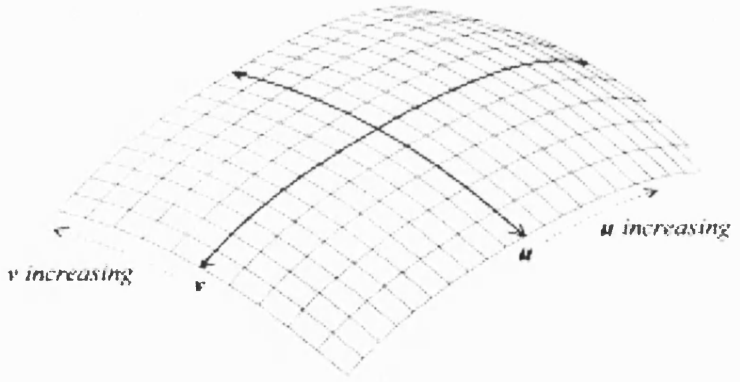


Figure 3.1.2c Diagram showing a system of surface co-ordinates

Spiral Geometry

This section introduces the spiral as a tool for design, and in particular it introduces computer methods for generating spiral geometry. Spiral geometry reconciles the conflict in geometry on the British Museum roof described in Appendix D of this dissertation.

A spiral is a curve, which rotates about a fixed point with an ever-increasing radius. Spirals are a recurrent theme of nature. Their underlying rules preoccupied the minds of ancient thinkers. Heath (1949) recalls Iamblichus's discussion of attempts by ancient thinkers to square the circle, that is, to find a square of equivalent area to a circle. Pythagoras studied the problem, as did Archimedes and Nicomedes, who both devised solutions using a spiral-shaped curve, which Nicomedes called a *quadratrix*.

D'Arcy Thompson (1961) devotes a chapter to the subject of spirals, treating the mathematics at some length. Cook (1908) also discusses spirals in detail, but does not elaborate on their mathematical properties. He marvels at their persistence in the natural world as does Pettigrew.

The 'spiral stair', as commonly referred to, is in fact not a spiral at all. It is a helix whose curve has the property of maintaining a constant radius as it ascends. The distinction between a spiral and a helix is that whereas a spiral has an ever-increasing radius, a helix, according to Lord and Wilson (1984),

'is a curve which remains at a constant distance from a fixed line or axis and forms a constant angle with planes perpendicular to the axis'.

Dürer however, constructed a conical helix as shown in figure 3.1.2d, from a flat spiral.

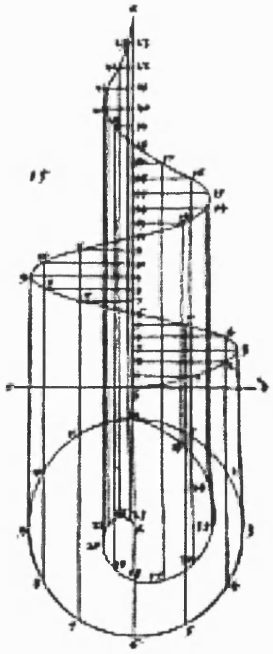


Figure 3.1.2d Dürer's construction of a conical helix from a flat spiral

Any spiral can be written

$$r = r(\theta)$$

in which r is the radius and θ is the angle between the radius and the x axis. The shape of the curve is determined by the function $r(\theta)$. In the case of an equable or Archimedean spiral, the radius increases at a steady rate, and the angle, α, between the radius and the tangent gradually changes, tending toward a right angle as a function of the increasing radius as seen in figure 3.1.2e.

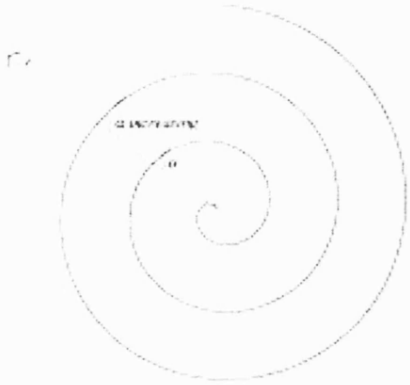


Figure 3.1.2e Spiral of Archimedes or Equable spiral

In the case of the logarithmic or equiangular spiral, which in botany is known as the ontogenetic spiral, the radius increases exponentially with the angle θ, whilst the angle α remains constant, see figure 3.1.2f.

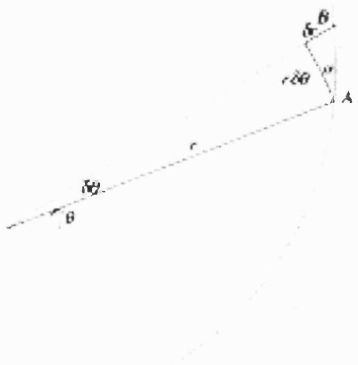


Figure 3.1.3f The Logarithmic spiral or Equiangular spiral

The geometry of a log spiral is given by,

$$r = r_o e^{\lambda\theta},$$

in which λ and r_o are constants. r_o is the value of r when θ is zero and e, is the exponential constant equal to 2.71828. . . .

Furthermore, for any curve the tangent of the angle α,

$$\tan\alpha = \frac{dr}{rd\theta}$$

if θ is measured in radians.

In the case of the logarithmic spiral, after substituting for r and then differentiating, we obtain,

$$= \frac{r_o \lambda e^{\lambda\theta}}{r_o e^{\lambda\theta}}.$$

Therefore, $\lambda = \tan\alpha$, and so if $\alpha = \frac{\pi}{4}$ then $\lambda = 1$. Note that 2π radians is equivalent to 360° so that $\frac{\pi}{4}$ is equivalent to 45°.

Spirals can be constructed from many geometrical constructions assembled from regular polygons. D'Arcy Thompson gives the example of spirals constructed from a system of squares and from a system of hexagons as shown in figure 3.1.2g and 3.1.2h.

Particularly pleasing spirals result from figures that increase or decrease gnomonically, that is, those figures whose proportions are maintained as they grow or diminish. D'Arcy Thompson gives a good description of this, when he remarks that,

'(1) if a growing structure be built up of successive parts, similar in form, magnified in geometrical progression, and similarly situated with respect to a centre of similitude, we can always trace through corresponding points a series of equiangular spirals; and (2) it is characteristic of the growth of the horn, of the shell, and of all other organic forms in which an equiangular spiral can be recognised, that each successive

increment of growth is similar, and similarly magnified, and similarly situated to its predecessor, and is in consequence a gnomon to the entire pre-existing structure. Conversely (3) it follows that in the spiral outline of the shell or of the horn we can always inscribe an endless variety of other gnomonic figures, having no necessary relation, save as a mathematical accident, to the nature or mode of development of the actual structure.'

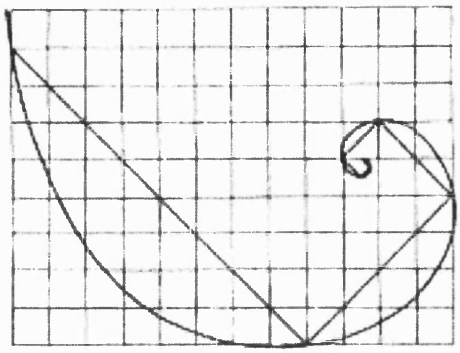


Figure 3.1.2g A spiral constructed from a system of squares

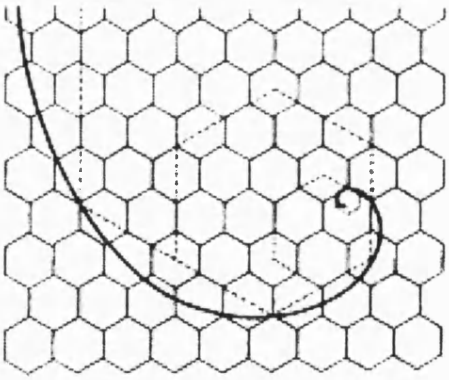


Figure 3.1.2h A spiral constructed from a system of hexagons

The Golden Section

The Golden Section ratio offers one such opportunity to construct a well-proportioned spiral and has been so used for centuries. The Golden Section ratio formed the basis of Le

Corbusier's *Modulor* and evidence suggests that the Ancient Greeks and Romans used it as a governing device for their proportioning systems in architecture.

The Golden Section is derived from principles of 'divine ratio', which ancient philosophers and mathematicians believed existed in natural geometry. Indeed some mollusc shells conform exactly to the Golden Section ratio.

Sharp (1997) illustrates ways in which the Golden Section rectangle is used to show spiral similarity and he discusses further links to the Fibonacci series. He suggests the possibility of constructing spiral geometry from the Golden Section rectangle by extracting 'whirling squares', a term that Jay Hambidge uses in his book about the Golden Section.

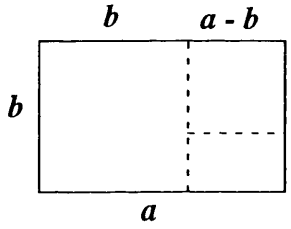


Figure 3.1.2i The Golden Section rectangle

Consider figure 3.1.2i where diminishing reciprocal rectangles are created by extracting squares formed from the short sides of each new rectangle. The ratio of the sides of any one of these rectangles is consistent with all the others if the following relationship is upheld:

$$\frac{a-b}{b} = \frac{b}{a},$$

from which we obtain,

$$a^2 - ab = b^2,$$

and

$$\frac{a^2}{b^2} - \frac{a}{b} - 1 = 0.$$

Solving the quadratic, we obtain,

$$\frac{a}{b} = \frac{1+\sqrt{5}}{2} = 1.6180339 \ldots ..$$

which is the value of the Golden Section ratio. In order to construct a Golden Section Spiral, this ratio is made equal to the ratios of radii when rotated through 90 degrees or $\frac{\pi}{2}$. Therefore we have

$$\frac{1+\sqrt{5}}{2} = e^{\frac{\lambda\pi}{2}},$$

so that,

$$\frac{\lambda\pi}{2} = \log\left(\frac{1+\sqrt{5}}{2}\right),$$

and,

$$\alpha = \tan^{-1}\left[\frac{2}{\pi}\log\left(\frac{1+\sqrt{5}}{2}\right)\right] = 0.29727 \ldots$$

which is 17.03°.

These relationships can be used to write a computer program to produce the Golden Section Spiral as presented below, and as illustrated in figure 3.1.2j.

Mathematical rules to Generate a Golden Section Spiral

The program used to generate the spiral illustrated in figure 3.1.2j is given in Appendix A1.1. Appendix C1.1 explains in detail, the structure and syntax of a simple computer program written in C++, using the program to generate the schematic geometry of the Rest Zone as an example. This appendix should be used as a guide to understanding the program in Appendix A1.1.

The mathematical rules used to calculate the co-ordinates of points, which result in a drawing of the Golden Section Spiral, are given in the boxes below. The rest of the program, which deals mainly with setting up the program and creating a *dxf* file, and which would remain unaltered if a different shape curve were to be drawn, appears in appendix A1.1.

```
lambda=(2.0/PI)*log((1.0+sqrt(5.
0))/2.0);

phi=atan(exp(lambda*PI/2.0));
for(i=0;i<=NoSegments;i+=1)
{
theta=phi+(2.0*PI*i)/(1.0*Points
PerCycle);
```

```
r=r0*exp(lambda*theta);
x[i]=r*cos(theta);
y[i]=r*sin(theta);
z[i]=0.0;
}
```

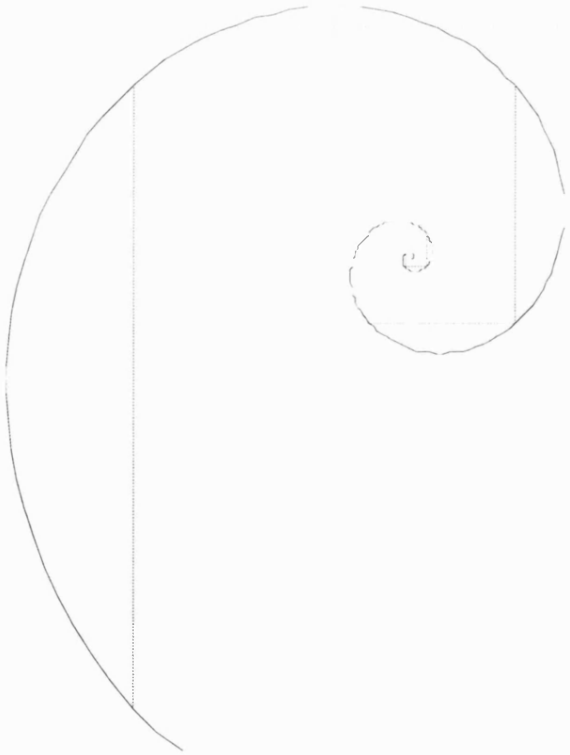


Figure 3.1.2j A spiral constructed from Golden Section proportions

Further programs, as well as their graphic output are given in Appendix A1.2 and A1.3. The first of these programs relates to the generation of Multiple Inter-locking Spirals based on system of curvilinear rectangles and curvilinear squares. The second relates to the generation of a 3-dimensional spiral or shell.

3.2 Complex Number Theory

The theory of complex numbers is described in various general and specialised mathematical texts, as well as in books on fluid mechanics and heat flow.

Fundamentally, complex numbers extend the possibilities of mathematics and enable solutions for all roots of quadratic, cubic and higher order polynomial equations to exist even though they may lie beyond the scope of the system of real numbers. They derive their name from Gauss, although Tartaglia, Cardano and Bombelli used them in the sixteenth century in their search to find solutions to cubic equations, which involved the roots of negative numbers.

Complex numbers are written in the form $x + iy$ or $a + ib$ where i is equal to $\sqrt{-1}$. In $x + iy$, x is referred to as the real part and iy is the imaginary part since $\sqrt{-1}$ does not exist.

Complex numbers are represented on an *Argand diagram,* shown in figure 3.2a, which exists on the complex plane.

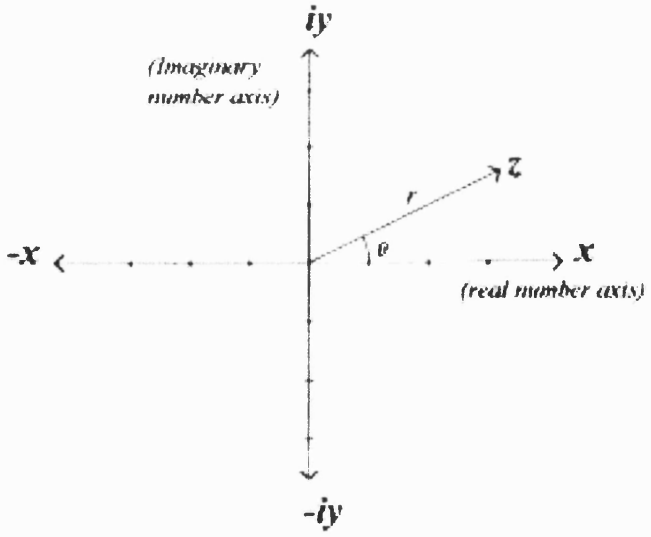


Figure 3.2a – Argand diagram

The rules governing their addition, subtraction and multiplication obey normal algebraic rules, their real and imaginary parts being computed separately. More detail on the algebra of complex numbers is given in appendix A2.0

Aside from their general mathematical applications, complex numbers have many other useful physical applications. They have particular uses in solving problems relating to heat flow, potential theory, fluid mechanics, electrostatics, aerodynamics and elasticity.

3.2.1 Complex Variables, Conformal mapping, Sources and Sinks

In this work, complex number theory is applied to the problem of generating fine curvilinear grids that resemble organic and inorganic structures, the diagrams resulting from magnetic force fields, the lines of flow emanating from a heat source or the patterns obtained from uniform flows travelling over rows of vortices.

The starting point for the studies is usually an orthogonal grid, which undergoes a deformation whereby its original co-ordinates are *mapped* to transformed co-ordinates using a mapping function. The transformation might be a translation, a displacement, a rotation, an inversion, an expansion or contraction or any combination of these.

Analytic functions of a complex variable are used here, because they are the only functions that will produce conformal maps, that is, maps in which the angles between lines are preserved. In particular curvilinear quadrilaterals are mapped to curvilinear squares, which have particular advantages when used for architectural applications.

They are termed analytic because they are continuous functions which are differentiable any number of times.

An analytic function of a complex variable, or *any* function of the form $\phi + i\psi = f(x + iy)$ produces a conformal map because functions of this type always results in,

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0 \text{ and } \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = 0,$$

i.e. ϕ and ψ both satisfy Laplace's Equation, the equation describing steady-state heat flow and irrotational flow of an incompressible fluid. Lines of ϕ indicate *equipotential* lines or lines of constant temperature and lines of ψ indicate *streamlines* or lines of flux.

D'Arcy Thompson explains that the art of creating conformal maps, which in his case was essentially a manual method, is an activity that is akin to the regular and

harmonious co-ordinates with which the physicist depicts the motions of a perfect fluid, or the theoretic field of force in a uniform medium.

By introducing the effect of a source or a sink, or indeed, a combination of several sources and sinks, each of which is assigned a strength, particularly interesting grids that have branching patterns result, as lines originating from one focal centre interact with curved lines emanating from other proximal centres.

Eduardo Catalano's idea for the structure of a city of life, circulation, growth and transformation uses a conformal map as the basis for its grid layout as shown in figure 3.2.1a.

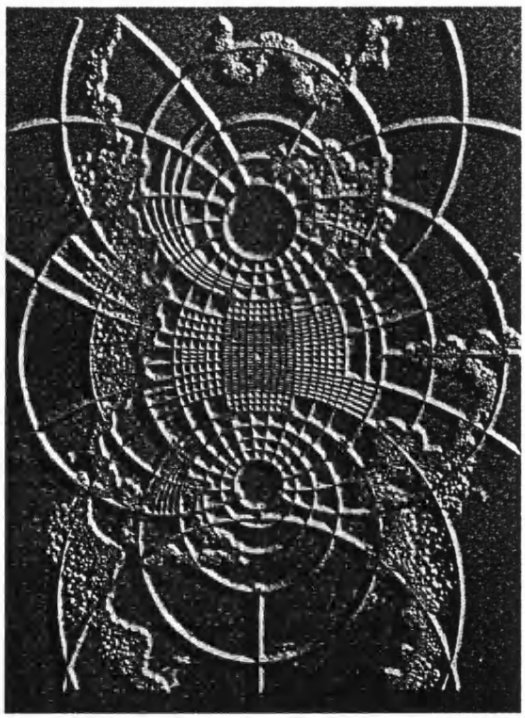


Figure 3.2.1a – Eduardo Catalano's model for a city system

Pier Luigi Nervi produced soffit designs in ferro-cement for the Gatti Wool Factory in Rome, 1953, and Palace of Labour, Turin, 1960, where the ribs were laid out according to the lines of principal stresses. The result is a web-like configuration of ribs that resembles a conformal map. Nervi believed that adhering strictly to the laws of statics made the most efficient use of materials, and that the fine aesthetic result was evidence of an affinity between physical laws and our own senses.

It will be shown in Chapter 4, in the design studies of bridges and spiral grids how these conformal maps can be given further architectural expression. Appendix A3.0 gives further examples of analytic functions of a complex variable and the conformal maps corresponding to these functions.

The programs that enable the co-ordinates of each map to be generated and processed into a line drawing are given after each map. The reader may find it useful to refer to appendix A2.0 for guidance on the algebra of complex numbers.

3.3 The Interface with Computers

There are several methods to select from in deciding how a curved form that exists in 3-dimensional space should be described or presented. Broadly speaking, the methods fall into three main categories - manual, computer-aided or mathematical. In practice, it is possible to combine any two or all three of these methods in the process of designing, presenting and making an object.

The approach adopted in this work emphasises a mathematical approach to form generation, the advantage being that small changes made to mathematical functions have an almost instantaneous effect on the shape of an object.

Faux and Pratt (1979) give a detailed account of the theory of computational geometry for design and manufacture with particular emphasis on the description of curved lines and surfaces using spline and other mathematical techniques. In addition, Lord and Wilson (1984) discuss the mathematical description of form in some detail, describing its underlying principles and giving examples of its application.

The purpose of this section is twofold; firstly, it aims to describe the development of one particular method of describing curved geometry – the spline method, which has been widely adopted for machine crafting in the automobile, marine and aerospace industries. The method evolved originally out of traditional techniques, and today combines a mathematical approach with computer processing technology.

No practical application of splines and b-splines has been made use of in this work, and so detailed examples of the mathematical functions that produce and modify this family of curves are not given here. However, spline techniques form the basis of many computer based drawing packages in the design industry, and as such, a general discussion in this work of its principles seems to be appropriate.

The second part of this section describes briefly the process through which data generated mathematically, is realised visually and physically using the computer as a processing medium. Further detail of *dxf* file content and structure is given in appendix A4.0.

3.3.1 Splines and B-splines

Traditionally, smooth curves were generated using a draughtsman's spline, which consists of a flexible bar bent around a series of pins. In passing from one pin to the next, the bar minimises strain energy, and generates curves known as elastica, which are similar in nature to cubics. The advantage of a spline curve is that for any given number of points, a smooth curve can be constructed joining them.

Most computer drawing packages contain the numerical equivalent of a spline curve, which is normally activated by specifying a series of control points. These control points replace the pins in a traditional spline, although the control points may not necessarily lie on the final curve.

From a graphical point of view, although a spline naturally creates a smooth curve linking a series of points, the visual appearance of the curve may not itself be satisfactory. This problem can be solved by adjusting the position of the pins, in the case of a draughtsman's spline, or by moving the control points, in the case of a computer-aided drawing.

The main disadvantage of the computer-aided description of spline curves and surfaces lies in the manual nature of its method, in the sense that the computer can only interpolate between points specified by the software operator. It does not generate forms or points of itself but instead relies on this information being supplied. The incorporation of changes can sometimes be a tedious procedure, since it is possible for a small geometrical change to either be accompanied by a large number of alterations to numerical data, or by the re-setting of a complete set of control points.

With regard to the mathematical generation of spline curves, there are a number of variants, three of which are discussed here.

Quadratic and cubic splines

A quadratic spline is one that is controlled by a second order function, and has continuity of slope alone, whilst a cubic spline is controlled by a third order function, and has continuity of both slope and curvature; in other words, it has continuous first and second derivatives. Although cubic splines can be constructed over any number of spans or

control points, one disadvantage is that local adjustments to one control point affect the entire spline geometry and positions of all its control points.

Aside from second and third order functions, spline geometry can also be generated from higher order functions, so that the **quartic spline**, which has continuous first, second and third derivatives, is created from a fourth order function. In general, however, splines or other curves generated from cubic functions offer a sufficient degree of smoothness for most design applications.

Quadratic and cubic B-splines

An alternative to quadratic and cubic splines are quadratic and cubic B-splines, which allow local adjustments to be made to a curve. The advantage of the B-spline over the simple spline, is that straight sections can be combined with curved lines without compromising continuity of slope and curvature, at the point of transition. Consequently, B-spline curves have a wider application in curve fitting and design.

Bi-quadratic and bi-cubic B-splines

As the suffix implies, bi-quadratic B-splines or bi-cubic B-splines extend the principles of spline curves to a two-way system of lines, giving rise to curved surfaces, with each function controlling one of the two surface co-ordinates. The resulting surface is called a 'patch'. As with quadratic splines, patches generated from bi-quadratic functions have continuity of slope alone, whereas surfaces generated from bi-cubic B-splines have continuity of both slope and curvature between patches.

The diagram in figure 3.3.1a illustrates an undulating surface constructed from three **bi-quadratic** B-spline patches. Each patch is generated from nine control points, and patches that lie adjacent to each other share six control points.

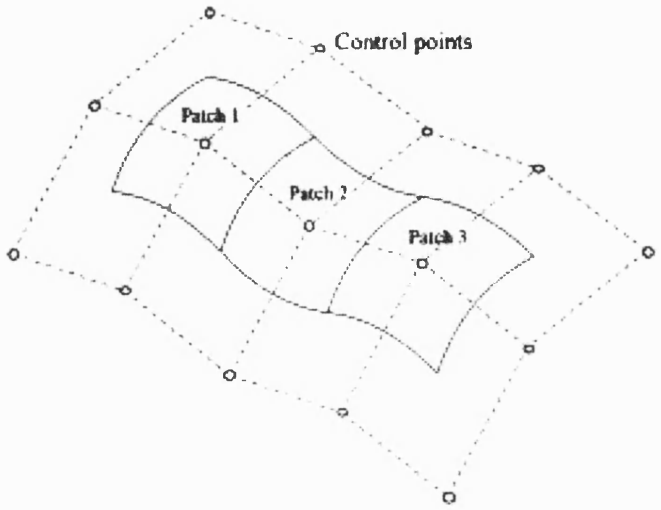


Figure 3.3.1a A bi-quadratic B-spline surface

In the case of a surface constructed from **bi-cubic** B-spline patches, each patch is generated from sixteen rather than nine control points, and adjacent patches have twelve control points in common. The higher the order of the function, the greater the degree of refinement, smoothness and continuity of the curved surface. In general, however, as with cubic splines, a bi-cubic B-spline offers sufficient refinement for most design purposes. Higher order functions become increasingly more difficult to control.

Bezier curves and surfaces work along much the same principles as cubic B-splines and bi-cubic B-splines.

3.3.2 Drawing Exchange Files

The multiplicity of graphic formats now available within the computer industry makes the *dxf* file an indispensable item. As a file interchange system, it enables the data generated from computer programs to be imported into graphics, analysis and manufacturing programs, thus ensuring that the maximum benefit is derived from the accuracy and detail of the original data.

A *dxf* file breaks down the graphic data of an object into a discrete set of text data, making it a convenient and universal language in which to transport information between different computer media. Thus, drawing information can be edited, viewed or modified in its presentation. The *dxf* file transfers the numerical co-ordinates of lines, poly-lines,

curves, polygons, associated text and drawing descriptions from one source, into other internal drawing representations.

Standard *dxf* file format is arranged in sections, each of which contains data of various types. In the method of form description used in this research, only the ENTITIES section of a typical *dxf* file is used because the data generated in the computer programs consists only of drawing entities, that is, lines, polylines, 3Dfaces, circles, etc., and does not contain any other information about the drawing. All other information such as text, view information etc. can be added in standard drawing package software.

The data types in a *dxf* file are assigned group codes and value types, which are instantly recognisable by other *dxf* compatible graphics software. Since this research is concerned with generating data to describe curved objects in terms of Cartesian co-ordinates, only the group codes that hold x, y and z co-ordinates are used in this work, together with a few other necessary support sections.

In standard *dxf* format, x co-ordinates are allocated the group codes between 10 and 18, y co-ordinates are allocated the group codes between 20 and 28, and z co-ordinates are allocated the group codes between 30 and 37. In this work, the first two group codes, i.e. $10-11, 20-21$ and 30 -31 for x, y and z co-ordinates are sufficient to store the data of lines, while the first four group codes of each co-ordinate type are sufficient to store the data of 3D faces. This is because, in general, data relating to lines has only two respective x, y and z co-ordinates each, the first set of co-ordinates describes the start point of the line and the second set describe the end point of the line. In the case of a 3D face, there are four sets of x, y and z co-ordinates, and each set describes the four corners of the 3D face.

In addition to the general sections, all *dxf* files have an end of file section, labelled, EOF.

More details on the content and structure of a *dxf* files are given in Appendix 4.0. Also in Appendix 4.0, is a computer program developed for the Rest Zone design study, which is used to illustrate how a typical instruction in C++ is written to create an output file in *dxf* format. Following this is a listing of the actual content of the *dxf* file generated by this C++ instruction.

Chapter 4.0 The Design Studies

This chapter describes the process of creating architectural and structural form using the language of mathematics. The mathematical principles introduced in chapter 3, combined with computer programming techniques, are used to produce a series of designs, which explore the structural and geometric characteristics of a range of natural forms.

Inspiration is drawn largely from the stiff structures of bones and shells, the curved geometry of spiral formations, the radial and filigree geometry of unicellular marine organisms and the branching and membranous networks of trees, spider webs and wing structures, examples of which have been presented in chapter 2. The first study deals with branching methods, which were developed in response to the briefs for 3 design competitions. This study uses complex analysis to generate curvilinear grids. The use of complex functions generates branching patterns, which are applied to various design problems.

The second study is based upon a real project, which has been built. It considers mathematical methods to generate the geometry of the Rest Zone for the Millennium Dome. The initial geometry-finding process uses conventional trigonometric methods to describe its schematic outline as a torus, which was then deformed to produce the final shape as built.

A third study, in which the author had only limited involvement, is discussed in Appendix D1.0. The process of creating a steel grid shell roof over the British Museum Great Court is described, where spiral geometry is used to merge the incompatibilities between the circular Reading Room and the rectangular Court perimeter.

The advantage of mathematical methods of generating form is that design and manufacture are integrated. The numerical data arising from the methods makes processing by structural analysis packages and automatic machining possible, leading to greater efficiency and accuracy in fabrication. However, the adoption of mathematical and computer processing methods does not, and should not render manual methods obsolete.

In most cases, the methods are best suited to the description of homogeneous objects, or objects that form a cohesive whole, which means that a single, concise computer

program provides enough information to generate the entire object. It is argued that the method is therefore organicist, since the morphology of the entire object is controlled by its genetic design code, the computer program.

The properties of continuity and change, which are features of many natural organisms, are used in all three studies to complete the construction of the entire object.

4.1 The Bridge and Wall Projects

The bridge projects explore the borderline between architecture and engineering, and more particularly, they explore the *aesthetics* of structure. An equal balance of aesthetics, function and structure is normally associated with and demanded of architectural projects, and is now becoming an important criterion for bridge design.

4.1.1 The Bridge Projects

4.1.1.1 Road Bridge Study 1

The first bridge study was the subject of a design competition for a harbour crossing at Holes Bay in Poole, Dorset.

An initial bridge study was explored using a modified sine function to generate a series of diminishing sinusoidal arches. An illustration of the scheme is shown in figures 4.1.1.1a to 4.1.1.1c, and a brief account of the method used to produce it is given in the section entitled Outline Geometry.

This initial study was not pursued to its full conclusion because the mathematical function used to describe its geometry produced a line drawing, which lacked detail and 3-dimensionality.

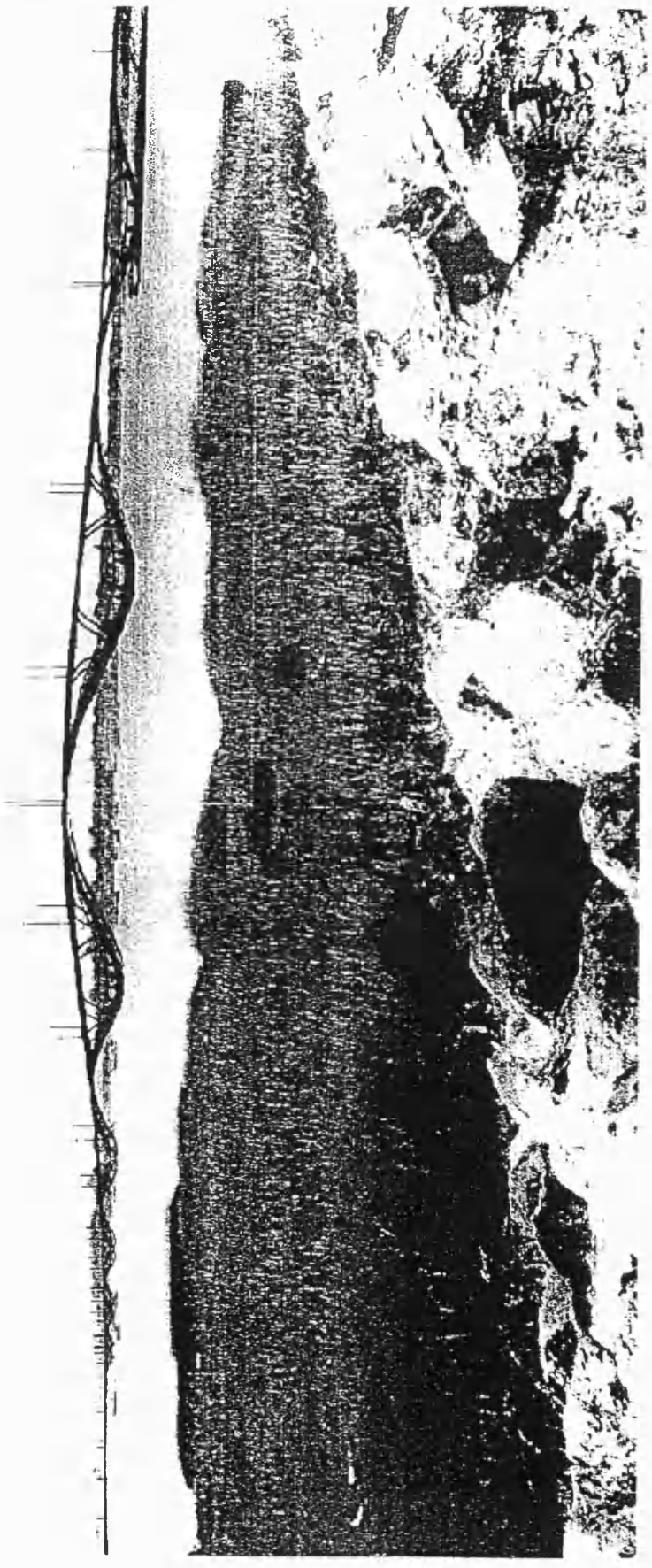


Figure 4.1.1.1a Persepective view of Roadbridge Study

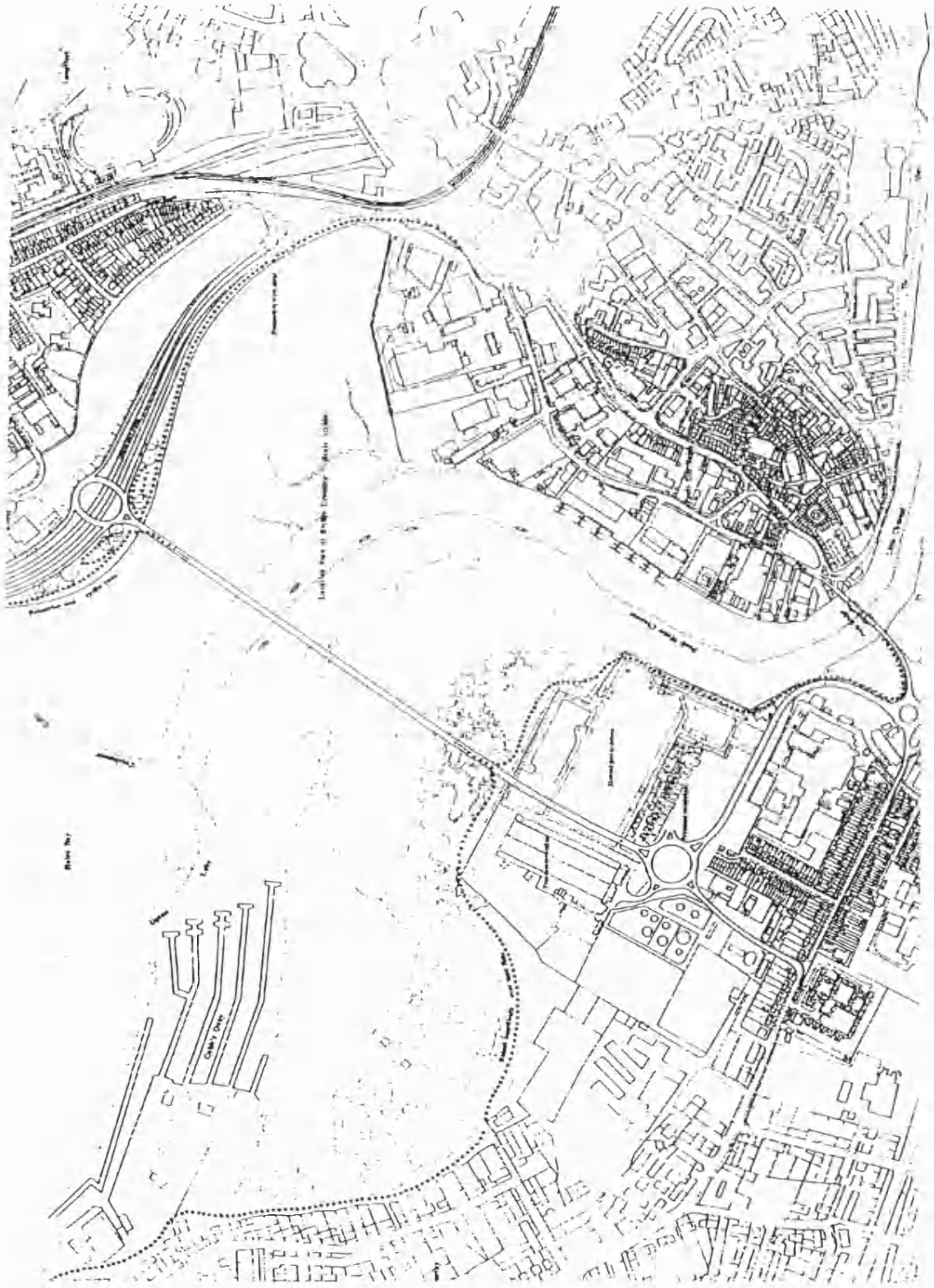


Figure 4.1.1.1b Location plan of Roadbridge Study

Figure 4.1.1.1c Graphed elevation of Roadbridge Study

Technical Constraints governing the bridge form

The brief for the bridge study required that equal consideration be given to technical, environmental and aesthetic requirements.

The technical constraints associated with the bridge crossing required a minimum clearance of 19.0m above High Water Medium Tide to be maintained. This clearance was to extend over a Maintained Channel width of 20m centred on the existing navigation channel.

A further technical requirement was that the extent below water of any structural elements, at any state of any tide, should not present a hazard to shipping.

The use of unpainted 'weathering steel' except as permitted by the Design Manual for Roads and Bridges was prohibited, as was the use of post-tensioned pre-stressed concrete construction with either bonded or internal un-bonded pre-stressing.

As a further structural constraint, any structure relying on tensioned cables for support in instances where those cables occurred above the level of the carriageway was required to be capable of remaining stable under full highway design loading, in the event of failure of one cable to carry loads.

All further general constraints were associated with Carriageway Design Standards and other traffic transport requirements.

Architectural and Environmental Objectives

The object of the architectural brief was to encourage an aesthetically appropriate design that would enhance the natural setting and promote the wildlife, ecosystems and activities of the area, both during construction and in the built condition. The aesthetic requirements sought to preserve the townscape and views from Poole Old Town.

The natural setting to the North is an area of low lying land of rural character, dominated by Upton Country Park, an area of pasture and woodland, designated Greenbelt and to the North-West, Pergin's Island, an unspoilt tree preservation and bird-watching site.

The specific aesthetic criteria required that consideration be given to **form, order, unity** and **artistic shaping.** In a word, an *organic* response.

Outline Geometry, Structure and Erection procedure

The aim behind the design of the Poole Harbour Bridge Crossing was to capture the dynamism of moving water. A series of low-lying sinusoidal arches creates the overall wave-like profile and acts as a natural complement to the low-lying land in the Holes Bay area.

The arches are located beneath the deck so as to minimise the impact of structural elements above it and to ensure that views looking from the Old Town of Poole towards the landscape of Pergin's Island and beyond, or from the north of the site towards the Old Town, are as far as possible maintained.

The arches are built out of parts that recall the neural spines of vertebrae. These parts make up a backbone, which gently skims the surface of Holes Bay. Initially, inspiration for the bridge was drawn from the *Pont des Arts* in Paris. The height of the roadway was determined by the conditions where the bridge meets the land and by the minimum clearance above the navigation channel. This lead to the varying height, and therefore the span of each arch was chosen to maintain the proportions of the arches. The single equation

$$y = \frac{c \log\left[2 + \cos\left\{b\left(\frac{x}{a} + \frac{1}{3}\frac{x^3}{a^3}\right)\right\}\right]}{\left(1 + \frac{x^2}{a^2}\right)\log 3},$$

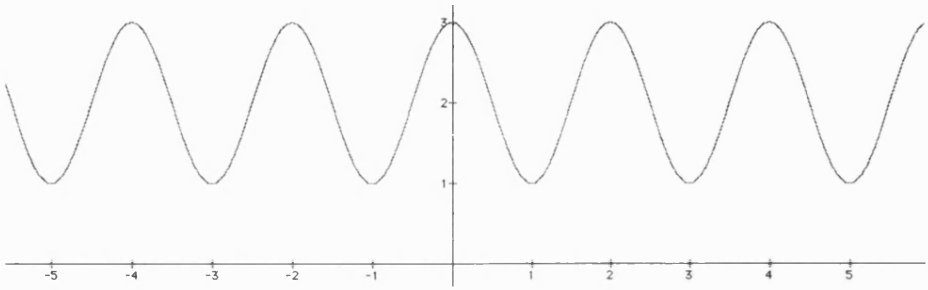
where *a, b* and *c* are constants, defines the shape and span of all the arches. Thus the arches have a 'pure' mathematical form free from discontinuities.

The equation was chosen to describe approximately parabolic arches that only deviated significantly as the supports were approached. The deviation causes a concentration of bending moment, which is resisted by the combination of the arch in compression and tension in the member above the arch.

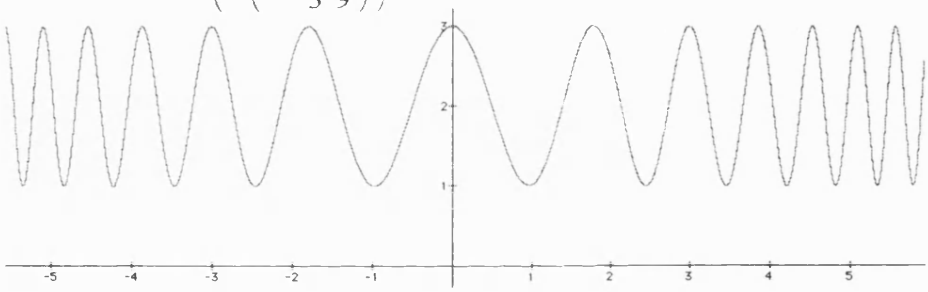
The mathematical function was developed through a process of trial and error, by using a graph-sketching calculator to visualise the effect of small changes made at each

stage of the function-finding process. The result of this exercise was then translated into programming code. Figure 4.1.1.1.d shows an example of this graphing calculator work.

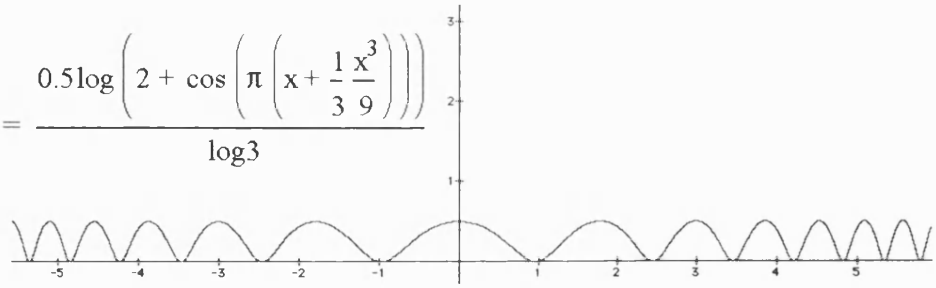
$$y = 2 + \cos(\pi x)$$



$$y = 2 + \cos\left(\pi\left(x + \frac{1}{3}\frac{x^3}{9}\right)\right)$$



$$y = \frac{0.5\log\left(2 + \cos\left(\pi\left(x + \frac{1}{3}\frac{x^3}{9}\right)\right)\right)}{\log 3}$$



$$y = \frac{0.5\log\left(2 + \cos\left(\pi\left(x + \frac{1}{3}\frac{x^3}{9}\right)\right)\right)}{\left(1 + \frac{x^2}{9}\right)\log 3}$$

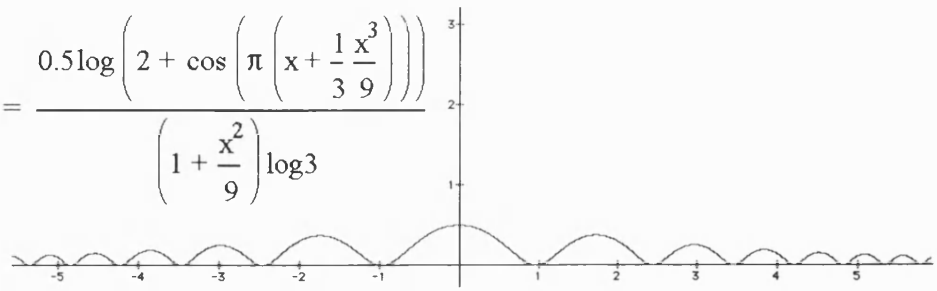


Figure 4.1.1.1.d Graphing Calculator Work

The erection procedure anticipated a *growth* process that worked from a 'floppy' structure composed of segments that would achieve its final geometry and load-bearing capability through post-stressing of the tension member above the vertebral units, after the V-struts had been installed into position. The erection process would commence at each support condition by seating the centre member onto the support piles first, and then by working away from the supports and towards the crown of the arches.

After stressing the tension member to develop the correct shape, the deck would then be placed in sections over the supporting skeleton. A high degree of geometrical accuracy both during the production information and fabrication stage was required such as would have an impact on the speed and ease of erection and therefore on the final visual result.

Given the chosen method for producing the curvilinear geometry, a large degree of control was possible, as were adjustments to the shape during the design process. Although the computer method in itself enabled the generation of a smoothly curved form that would have been difficult to describe fully and accurately manually, the lack of 3-dimensionality of the chosen technique made it difficult to pursue this method to its conclusion.

A modified scheme was therefore pursued using branching methods, as described in the next section.

4.1.1.2 Road Bridge Study 2

Outline Geometry

The modified road bridge study is illustrated in figure 4.1.1.2a. The method used to define its geometry uses a complex analytic function incorporating an array of three sources.

Lines of constant ϕ correspond to 'contour lines' shown on the conformal map in figure 4.1.1.2b. Extruding the so-called contour lines and then distorting them gives rise to the cross-sections of the bridge shown in figure 4.1.1.2c. Because each of the lines of constant ϕ is a closed line, extruding it creates a 'tube', or 3-dimensional member.

The cross-section at any point is controlled by the location of the three sources, that is the values of y at the centre of each source, and by the value of ϕ on that cross-section.

As these values change from cross-section to cross-section individual members may merge or a single member may branch into two.

Figure 4.1.1.2a Part elevation and detail of modified Road Bridge study

Circumferentials are lines of constant ϕ
Radials are lines of constant ψ

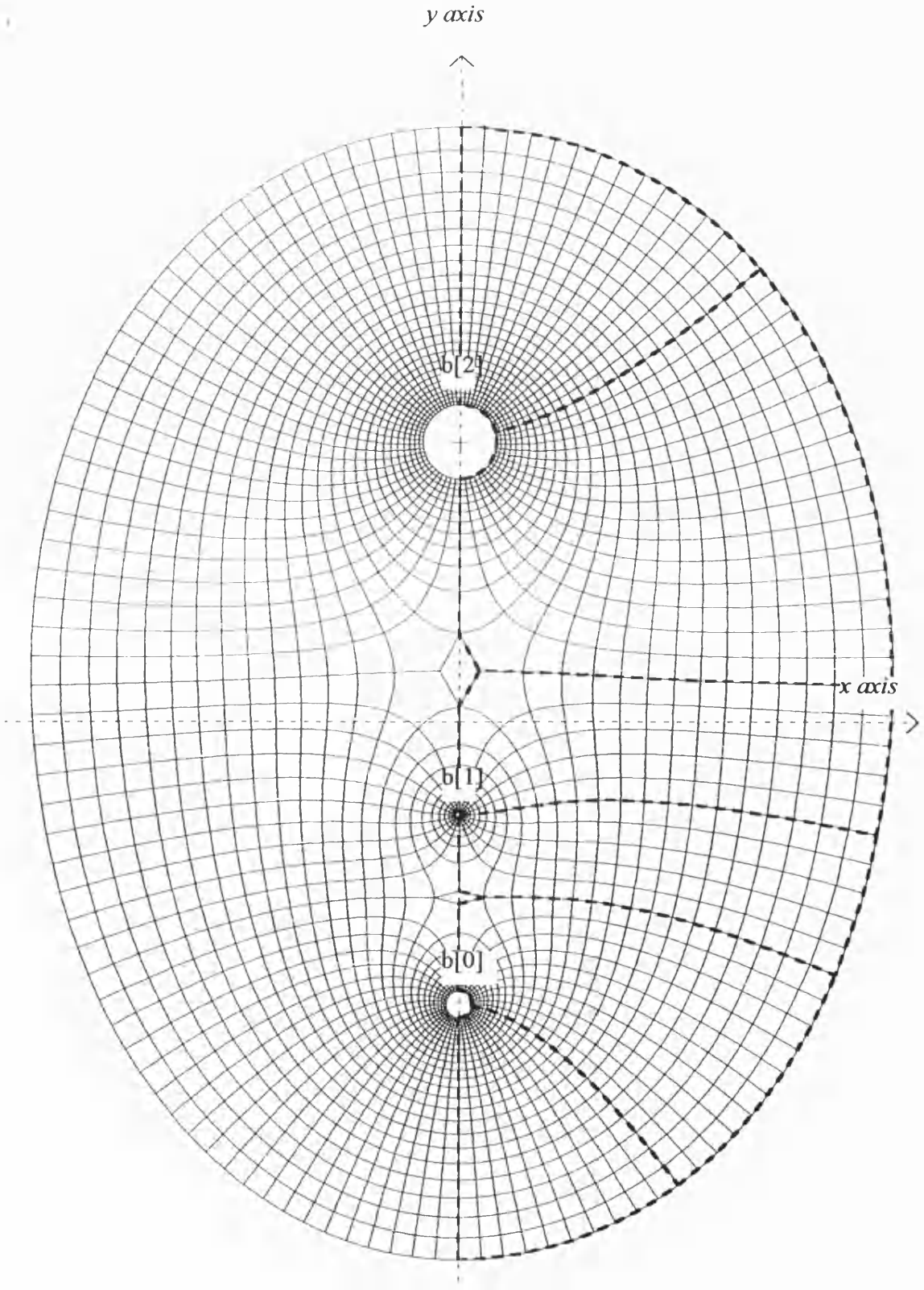


Figure 4.1.1.2b Conformal map study of three sources for Roadbridge

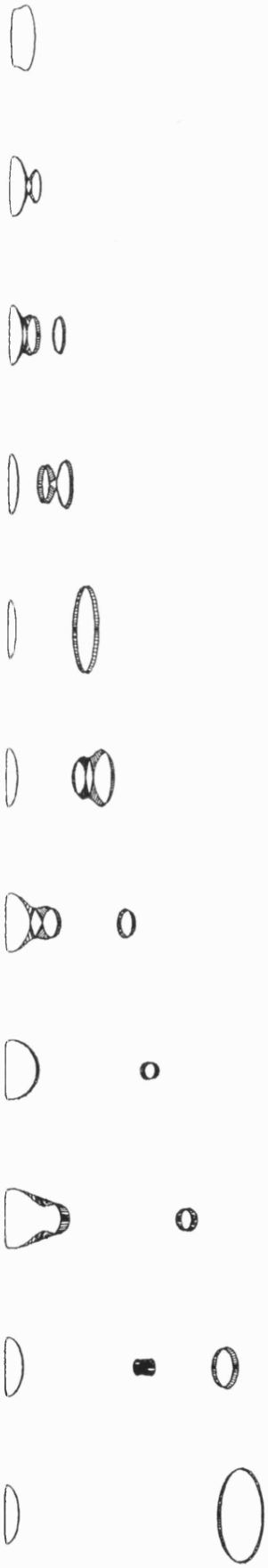


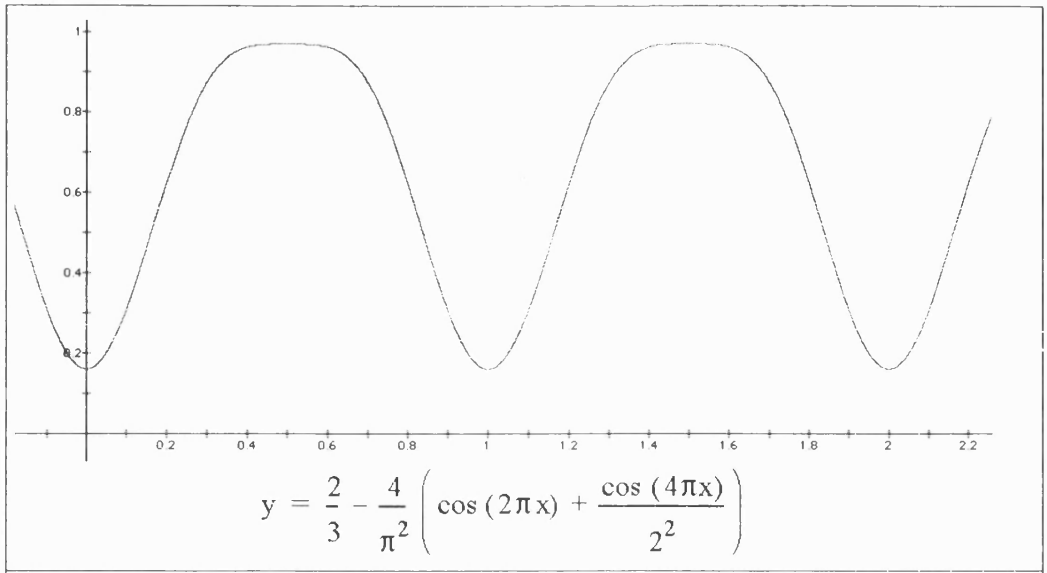
Figure 4.1.1.2c Cross-sections of the bridge created from the progression of sources

The location of the sources and the value of ϕ are controlled by different Fourier series of the longitudinal co-ordinate. Fourier functions are normally applied to harmonic analysis in acoustics, vibration theory, optics, signal analysis and quantum mechanics. They use sine and cosine functions to describe the waveform of sound and electrical signals.

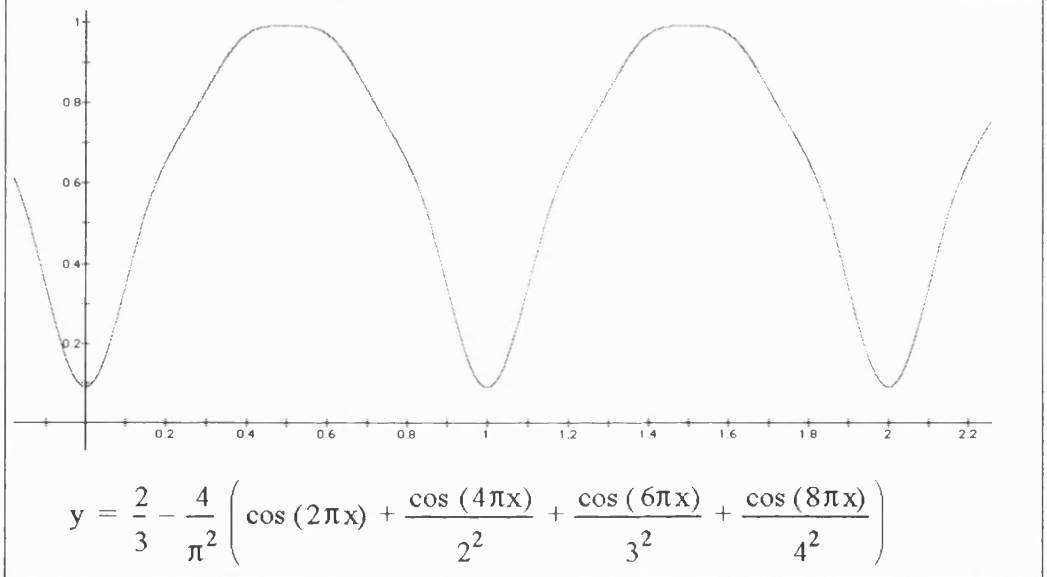
The shape of the arch is described by taking a limited number of terms of the Fourier series for a parabola,

$$\frac{y}{L} = \frac{2}{3} - \sum_{n=1}^{\infty} \frac{4}{n^2\pi^2} \cos\frac{2\pi n x}{L}$$

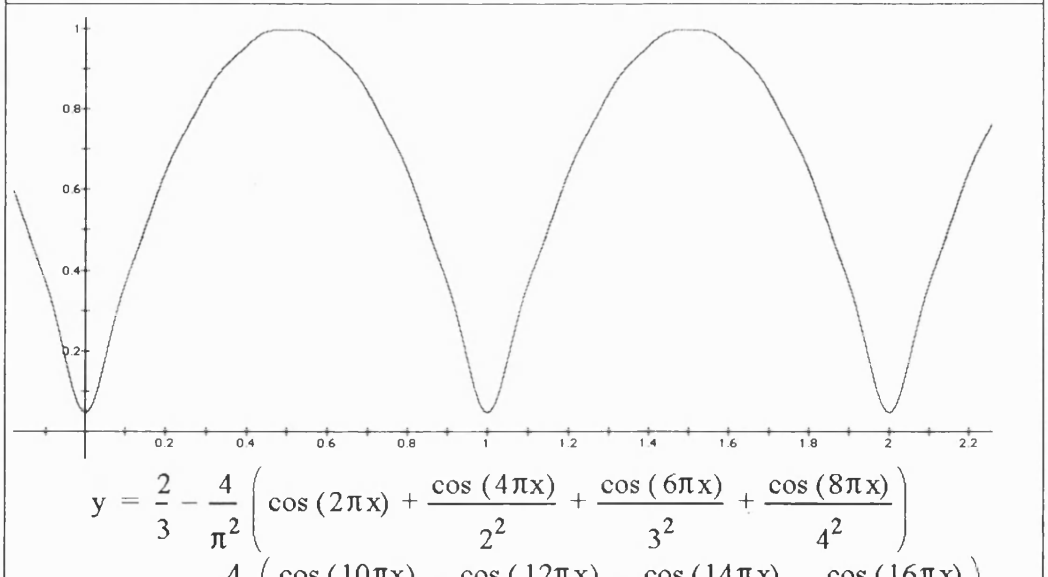
and figures 4.1.1.2d and 4.1.1.2e show the results from taking 2, 4, 8 or 16 terms of the series. The Fourier series with a limited number of terms 'rounds off' the kink where two parabolas meet.



$$y = \frac{2}{3} - \frac{4}{\pi^2}\left(\cos(2\pi x) + \frac{\cos(4\pi x)}{2^2}\right)$$

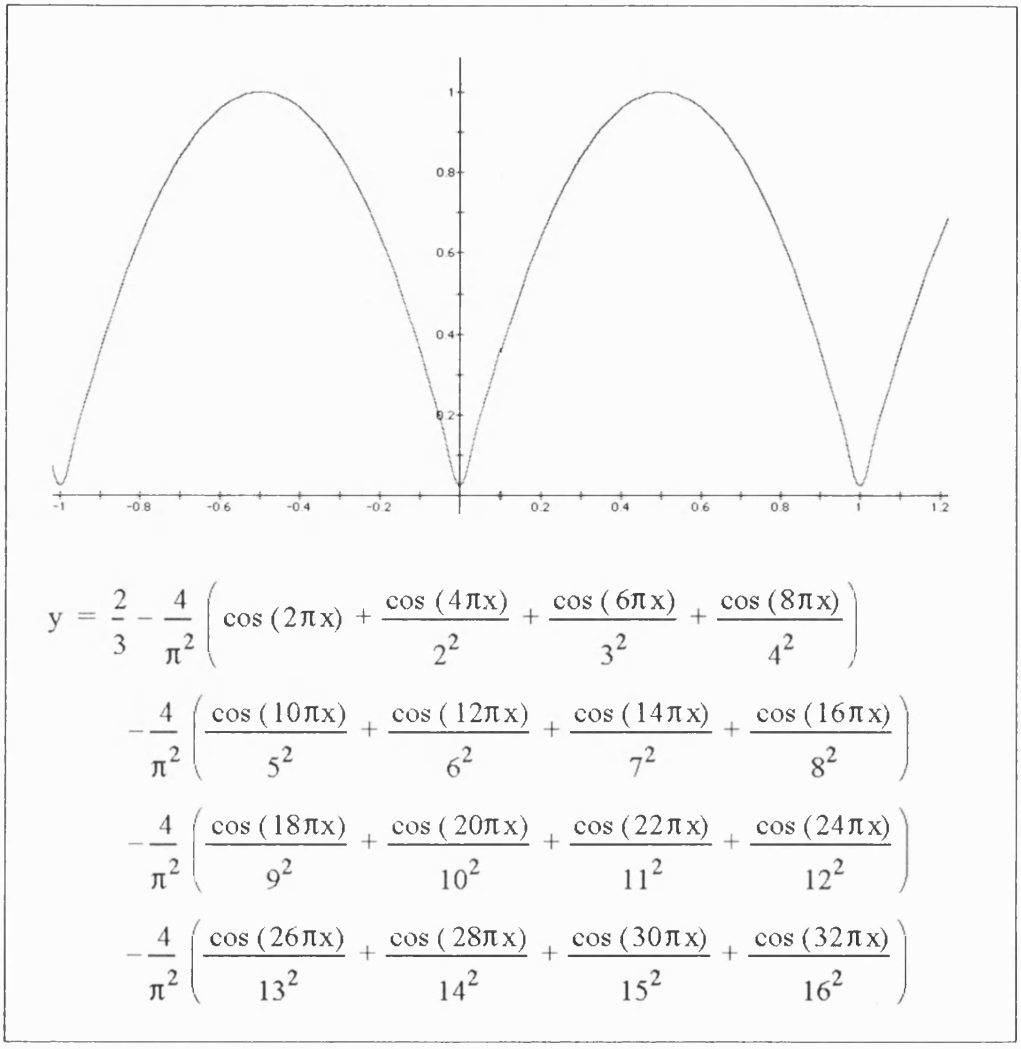


$$y = \frac{2}{3} - \frac{4}{\pi^2}\left(\cos(2\pi x) + \frac{\cos(4\pi x)}{2^2} + \frac{\cos(6\pi x)}{3^2} + \frac{\cos(8\pi x)}{4^2}\right)$$



$$y = \frac{2}{3} - \frac{4}{\pi^2}\left(\cos(2\pi x) + \frac{\cos(4\pi x)}{2^2} + \frac{\cos(6\pi x)}{3^2} + \frac{\cos(8\pi x)}{4^2}\right) - \frac{4}{\pi^2}\left(\frac{\cos(10\pi x)}{5^2} + \frac{\cos(12\pi x)}{6^2} + \frac{\cos(14\pi x)}{7^2} + \frac{\cos(16\pi x)}{8^2}\right)$$

Figure 4.1.1.2d Fourier series for a parabolic wave



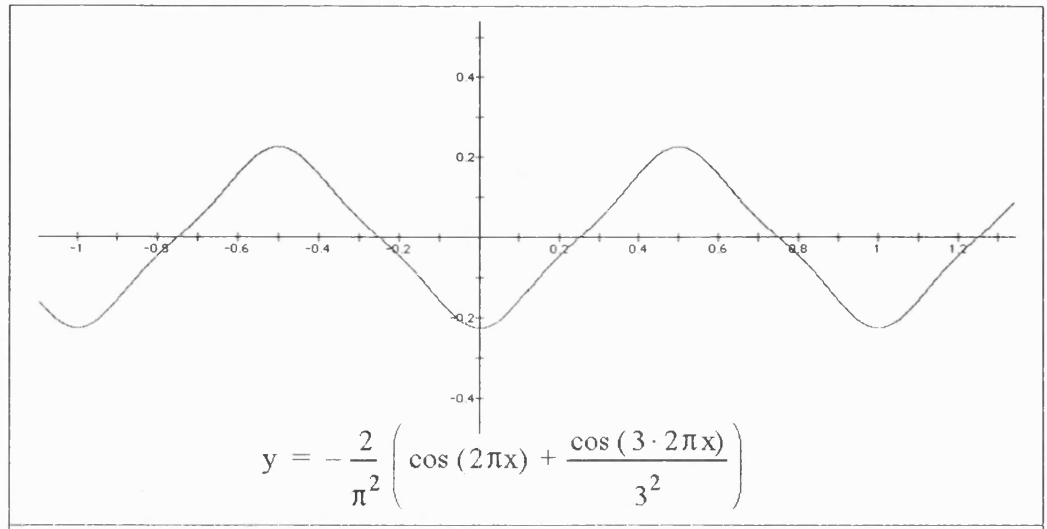
$$y = \frac{2}{3} - \frac{4}{\pi^2}\left(\cos(2\pi x) + \frac{\cos(4\pi x)}{2^2} + \frac{\cos(6\pi x)}{3^2} + \frac{\cos(8\pi x)}{4^2}\right)$$
$$-\frac{4}{\pi^2}\left(\frac{\cos(10\pi x)}{5^2} + \frac{\cos(12\pi x)}{6^2} + \frac{\cos(14\pi x)}{7^2} + \frac{\cos(16\pi x)}{8^2}\right)$$
$$-\frac{4}{\pi^2}\left(\frac{\cos(18\pi x)}{9^2} + \frac{\cos(20\pi x)}{10^2} + \frac{\cos(22\pi x)}{11^2} + \frac{\cos(24\pi x)}{12^2}\right)$$
$$-\frac{4}{\pi^2}\left(\frac{\cos(26\pi x)}{13^2} + \frac{\cos(28\pi x)}{14^2} + \frac{\cos(30\pi x)}{15^2} + \frac{\cos(32\pi x)}{16^2}\right)$$

Figure 4.1.1.2e Fourier series for a parabolic wave

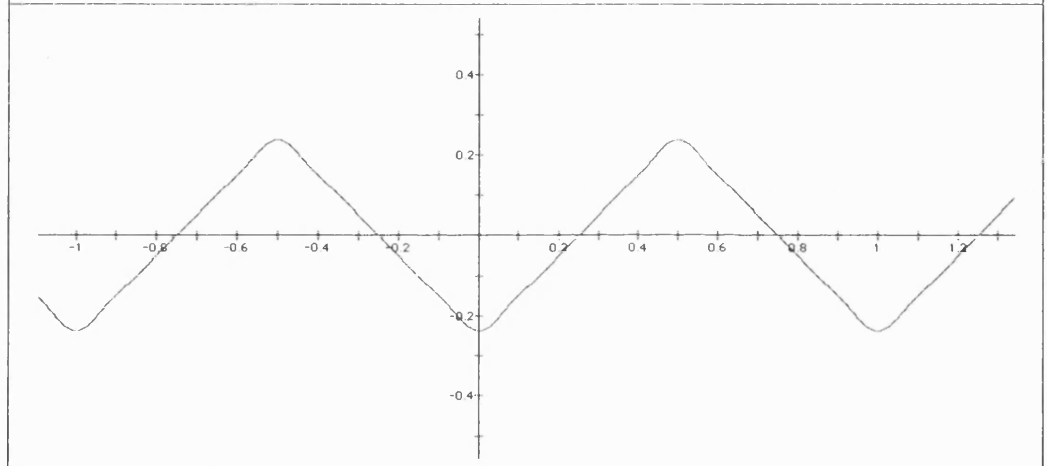
The diagonal bracing between the arch and deck is described by taking a limited number of terms of the Fourier series for a triangular wave,

$$y = -\frac{2L}{\pi^2}\sum_{n=1}^{\infty}\frac{\cos\left((2n-1)\frac{2\pi x}{L}\right)}{(2n-1)^2}$$

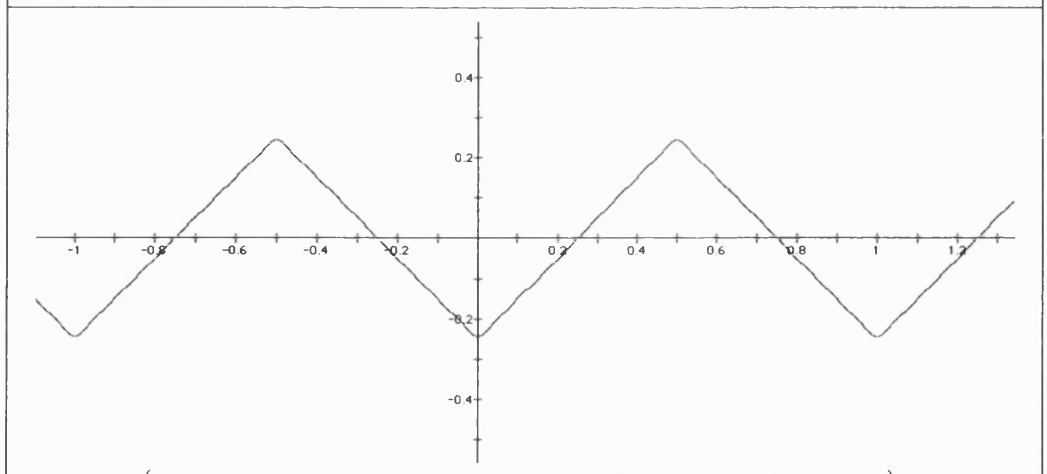
and figure 4.1.1.2f shows the results from taking 2, 4 or 8 terms of the series.



$$y = -\frac{2}{\pi^2}\left(\cos(2\pi x) + \frac{\cos(3\cdot 2\pi x)}{3^2}\right)$$



$$y = -\frac{2}{\pi^2}\left(\cos(2\pi x) + \frac{\cos(3\cdot 2\pi x)}{3^2} + \frac{\cos(5\cdot 2\pi x)}{5^2} + \frac{\cos(7\cdot 2\pi x)}{7^2}\right)$$



$$y = -\frac{2}{\pi^2}\left(\cos(2\pi x) + \frac{\cos(3\cdot 2\pi x)}{3^2} + \frac{\cos(5\cdot 2\pi x)}{5^2} + \frac{\cos(7\cdot 2\pi x)}{7^2}\right)$$
$$-\frac{2}{\pi^2}\left(\frac{\cos(9\cdot 2\pi x)}{9^2} + \frac{\cos(11\cdot 2\pi x)}{11^2} + \frac{\cos(13\cdot 2\pi x)}{13^2} + \frac{\cos(15\cdot 2\pi x)}{15^2}\right)$$

Figure 4.1.1.2f Fourier series for a triangular wave

The program used to generate this design study is given in appendix B1.0. The mathematical principles that underlie the method of calculating the co-ordinates of the 2-dimensional map are explained below.

Detailed geometry-finding

Consider the complex number z, whose real and imaginary parts are the Cartesian co-ordinates, x and y,

$$z = x + iy \qquad (1)$$

and a second complex number, η, whose real and imaginary parts are given by

$$\eta = \phi + i\psi \ . \qquad (2)$$

The function

$$\eta = A \log \frac{z}{c} \qquad (3)$$

where A and c are constants describes a source at the origin, around which there are circular lines of constant ϕ, called *equipotentials* (see section 3.2.1) and radial lines of constant ψ called *streamlines* or *flowlines*. Note that c, x, y and z have the units of length while A, ϕ, ψ and η are dimensionless. The constant A is the *strength* of the source and c controls the scale.

Similarly, the function,

$$\eta = -A \log \frac{z}{c}, \qquad (4)$$

describes a sink. Sources and sinks are essentially the same, except that the direction of flow is reversed - a sink is a source with a negative strength. In (3) ϕ increases with distance from the origin, while in (4) it decreases and some authors would interchange (3) and (4).

The mapping function (3) produces

$$\phi + i\psi = A\log\frac{x+iy}{c}, \qquad (5)$$

but in order to produce the mapping pattern given in figure 4.1.1.2g, the problem involves finding the co-ordinates of x and y that correspond to particular values of ϕ and ψ.

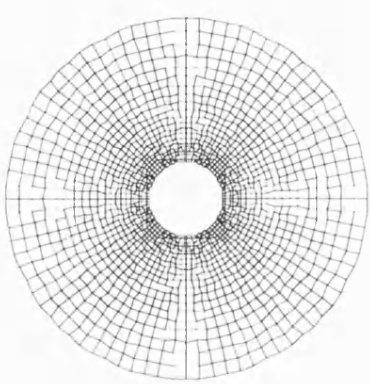


Figure 4.1.1.2g Conformal map study of a single source

The relationship in (5) above can be inverted to give $x+iy$ in terms of $\phi+i\psi$,

$$\begin{aligned} x+iy &= ce^{\left(\frac{\phi+i\psi}{A}\right)} \\ &= ce^{\frac{\phi}{A}}e^{i\frac{\psi}{A}}. \end{aligned} \qquad (6)$$

Using the rule from equation (11) in appendix A2.1,

$$ce^{\frac{\phi}{A}}e^{i\frac{\psi}{A}} = ce^{\frac{\phi}{A}}\left(\cos\frac{\psi}{A} + i\sin\frac{\psi}{A}\right), \qquad (7)$$

so that,

$$x = ce^{\frac{\phi}{A}}\cos\frac{\psi}{A} \text{ and } y = ce^{\frac{\phi}{A}}\sin\frac{\psi}{A}. \qquad (8)$$

A logarithmic function of a complex variable always produces a source or sink pattern. Thus $\eta = A\log\dfrac{z-(a+ib)}{c}$ produces a source at $x=a$, $y=b$ and

$\eta = A_1 \log\frac{z-(a_1+ib_1)}{c_1} + A_2 \log\frac{z-(a_2+ib_2)}{c_2}$ produces a source of strength A_1 at $x = a_1$, $y = b_1$ and a source of strength A_2 at $x = a_2$, $y = b_2$. Similarly $\eta = A_1 \log\frac{z-(a_1+ib_1)}{c_1} - A_2 \log\frac{z-(a_2+ib_2)}{c_2}$ produces a combination of a source and a sink.

Let us consider the case of three sources located on the y axis at $y = b_0$, $y = b_1$ and $y = b_2$ as shown in figure 4.1.1.2b.

At large values of z, (that is when the absolute value of x or y is large compared with b_0, b_1 and b_2) the effect of the individual source reduces and all three sources are embraced by a continuous line of constant ϕ conversely, as one approaches the sources, branching patterns are created at the points where the lines of ϕ associated with each source become separated from one other.

These three sources are represented by

$$\eta = A_0 \log\left(\frac{z-ib_0}{c_0}\right) + A_1 \log\left(\frac{z-ib_1}{c_1}\right) + A_2 \log\left(\frac{z-ib_2}{c_2}\right) \qquad (9)$$

which can also be written

$$\eta = \log\left(\left(\frac{z-ib_0}{c_0}\right)^{A_0}\left(\frac{z-ib_1}{c_1}\right)^{A_1}\left(\frac{z-ib_2}{c_2}\right)^{A_2}\right). \qquad (10)$$

The strength of the sources, A_0, A_1 and A_2, is apparent from the number of lines of ψ radiating from each source.

If $A_0 = A_1 = A_2 = A$, (10) can be rewritten as $(z-ib_0)(z-ib_1)(z-ib_2) - c_0c_1c_2e^{\frac{\eta}{A}} = 0$ which is a cubic in z and as such has an analytic

solution[§]. However, if A_0, A_1 and A_2 are not all equal, there will in general not be an analytic solution.

Therefore Newton's method was used using the algorithm

$$z_{new} = z_{current} + \frac{\left(\eta_{required} - \eta_{current}\right)}{\left(\frac{d\eta}{dz}\right)_{current}} \qquad (11)$$

repeatedly where

$$\eta_{current} = A_0 \log\left(\frac{z_{current} - ib_0}{c_0}\right) + A_1 \log\left(\frac{z_{current} - ib_1}{c_1}\right) + A_2 \log\left(\frac{z_{current} - ib_2}{c_2}\right) \qquad (12)$$

and

$$\left(\frac{d\eta}{dz}\right)_{current} = \frac{A_0}{z_{current} - ib_0} + \frac{A_1}{z_{current} - ib_1} + \frac{A_2}{z_{current} - ib_2}. \qquad (13)$$

$\eta_{required}$ is the value of η for which the corresponding value of z is sought, $z_{current}$ is the current approximation for z and z_{new} is an improved approximation. This process is illustrated in figure 4.1.1.2h.

[§] **General solution of cubic:** Consider cubic $Z^3 + AZ^2 + BZ + C = 0$. Write $Z = z - \frac{A}{3}$ so that $\left(z - \frac{A}{3}\right)^3 + A\left(z - \frac{A}{3}\right)^2 + B\left(z - \frac{A}{3}\right) + C = z^3 + \left(B - \frac{A^2}{3}\right)z + \frac{2A^3}{27} - \frac{AB}{3} + C = 0$ or $z^3 = 3pz + 2q$ where $p = -\frac{1}{3}\left(B - \frac{A^2}{3}\right)$ and $q = -\frac{1}{2}\left(\frac{2A^3}{27} - \frac{AB}{3} + C\right)$. Writing $z = u + \frac{p}{u}$ gives $u^3 + \frac{p^3}{u^3} - 2q = 0$. Thus $u^6 - 2qu^3 + p^3 = 0$ or $u^3 = q \pm \sqrt{q^2 - p^3} = \frac{p^3}{q \mp \sqrt{q^2 - p^3}}$.

Finally, $z = \left(q + \sqrt{q^2 - p^3}\right)^{\frac{1}{3}} + \left(q - \sqrt{q^2 - p^3}\right)^{\frac{1}{3}}$.

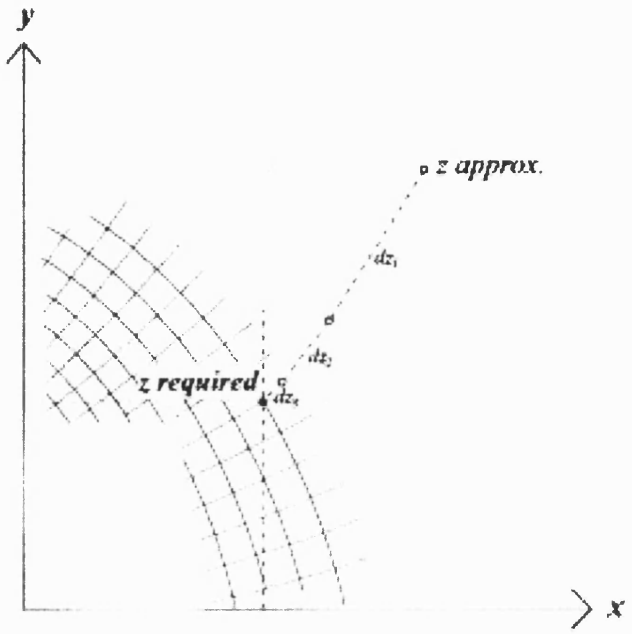


Figure 4.1.1.2h Diagram showing convergence path of z

As always with Newton's method, convergence is rapid when close to a solution, but it is always possible to get the wrong solution if one starts at the wrong place. The logarithm of the complex number $x+iy$ is equal to $\frac{1}{2}\log\left(x^2+y^2\right)+i\cos^{-1}\frac{x}{\sqrt{x^2+y^2}}=\frac{1}{2}\log\left(x^2+y^2\right)+i\sin^{-1}\frac{y}{\sqrt{x^2+y^2}}$ which is not unique in that multiples of $2\pi i$ can be added without changing x and y. Thus the main practical problem in computing figure 4.1.1.2b was the calculation of the appropriate value of η_{current} corresponding to z_{current} using equation (12).

The above discussion explains how a drawing such as figure 4.1.1.2b can be produced. In order to produce the cross-sections in figure 4.1.1.2c a number of procedures were used. Firstly the approximately circular equipotentials around each source were made elliptical by replacing the log functions in (9) by inverse hyperbolic cosines,

$$\cosh^{-1}\left(\frac{z-ib}{c}\right)=\log\left(\frac{z-ib}{c}+\sqrt{1+\frac{(z-ib)^2}{c^2}}\right). \qquad (14)$$

Conformal Map 3 in section 3.2.1 shows the form generated by a single inverse hyperbolic cosine. The validity of (14) can be demonstrated by taking the hyperbolic cosine of the right hand side to give

$$\cosh\left(\log\left(\frac{z-ib}{c}+\sqrt{\frac{(z-ib)^2}{c^2}-1}\right)\right)$$

$$=\frac{\dfrac{z-ib}{c}+\sqrt{\dfrac{(z-ib)^2}{c^2}-1}+\dfrac{1}{\dfrac{z-ib}{c}+\sqrt{\dfrac{(z-ib)^2}{c^2}-1}}}{2}$$

$$=\frac{\dfrac{z-ib}{c}+\sqrt{\dfrac{(z-ib)^2}{c^2}-1}}{2}$$

$$+\frac{\dfrac{z-ib}{c}-\sqrt{\dfrac{(z-ib)^2}{c^2}-1}}{2\left(\dfrac{z-ib}{c}+\sqrt{\dfrac{(z-ib)^2}{c^2}-1}\right)\left(\dfrac{z-ib}{c}-\sqrt{\dfrac{(z-ib)^2}{c^2}-1}\right)}$$

$$=\frac{\dfrac{z-ib}{c}+\sqrt{\dfrac{(z-ib)^2}{c^2}-1}}{2}+\frac{\dfrac{z-ib}{c}-\sqrt{\dfrac{(z-ib)^2}{c^2}-1}}{2}$$

$$=\frac{z-ib}{c}.$$

The second set of procedures to produce figure 4.1.1.2c from figure 4.1.1.2b involve the change in the values ϕ, of and of b_0, b_1 and b_2. The value of ϕ on a cross-section is

$$\phi=-3\pi-6\pi\frac{\left(1-\cos\dfrac{2\pi z}{L}\right)}{2},$$

where z is now the longitudinal co-ordinate, perpendicular to x and y in the plane of the cross-section. L is the span.

The value of b_2, corresponding to the deck, is kept constant, while

$$b_0 = b_2 - 16000\frac{4}{\pi^2}\left(\frac{\pi^2}{12} + \frac{\cos\left(\frac{2\pi z}{L}\right)}{1^2} + \frac{\cos\left(2\frac{2\pi z}{L}\right)}{2^2} + \frac{\cos\left(3\frac{2\pi z}{L}\right)}{3^2} + \frac{\cos\left(4\frac{2\pi z}{L}\right)}{4^2} + \frac{\cos\left(5\frac{2\pi z}{L}\right)}{5^2}\right),$$

and

$$b_1 = \frac{b_2 + b_0}{2} - \frac{(b_2 - b_0)}{\pi^2}\left(\frac{\cos\left(\frac{4\times 2\pi z}{L}\right)}{1^2} + \frac{\cos\left(3\frac{4\times 2\pi z}{L}\right)}{3^2} + \frac{\cos\left(5\frac{4\times 2\pi z}{L}\right)}{5^2} + \frac{\cos\left(7\frac{4\times 2\pi z}{L}\right)}{7^2} + \frac{\cos\left(9\frac{4\times 2\pi z}{L}\right)}{9^2}\right)$$

to correspond to the Fourier series for the parabolic and triangular waves discussed in the last section.

The 'waisting' or bone-like appearance of the members between nodes happens automatically.

4.1.1.3 The Footbridge Study

Technical Constraints governing its form

The footbridge study was the subject of a design competition for a pedestrian crossing over the Thames to connect St. Paul's Cathedral in London with the new Tate Gallery of Modern Art at Bankside.

As with the road bridge, the footbridge was required to provide navigational clearance. This clearance was not to be less than 7.1m above Mean High Water Springs in order not to prevent the passage of river traffic in compliance with the Port of London Authority and Environment Agency Thames Region requirements. The structure itself was to be no higher than 15.84m at the centre of the river so as not to impair views from and towards St. Paul's Cathedral. In addition, a flood defence level clearance of 5.41m AOD

was required at either side of the river, and any foundations or bridgehead structures were to take account of the archaeological, ecological and historical value of the site.

As in the case of the Poole study, the two opposing constraints of sight line and navigational clearance dictated the topside and underside limits within which the form of the bridge could be envisaged.

In addition, the structure was to comply with the relevant standards and design criteria set by the Department of Transport and the requirements and recommendations of appropriate British Standards including criteria for disabled access.

Architectural Objectives

The primary objective was to create a link between St. Paul's cathedral and the proposed New Tate Gallery of Modern Art, by means of a landmark feature that would reinforce London's status as a world-class city. The bridge was also intended to improve accessibility and encourage the use of the riverbank and surrounding environment.

The design and aesthetic criteria related to the requirement for a bridge of exceptional quality in terms of both design and engineering, and a solution that would maintain views along the river and towards St. Paul's Cathedral, whilst providing a vantage point to afford new views for the benefit of tourists and visitors.

Due to a preference by the Port of London Authority to keep clear the Navigable river and limit the area allowable for bridge piers to regions that fell outside the central and foreshore zone of the river, it was decided to omit bridge piers altogether and instead rely on the shoulders of the riverbank for support, bearing in mind the requirement for flood defence level clearance.

In bearing in mind these constraints, the analogy of a quadruped leaping or stretching to avoid a gap was adopted. Limiting the bridge geometry to a single span created the required form, however it would be possible to increase the number of spans by mirroring its outline. Doing this creates a form that resembles the wingspan of an albatross, which would be more appropriate to the design of a multiple-span roof canopy than to the design of a pedestrian bridge.

Structure, Materials, Construction and Erection

The bridge structure consists of a main arch span with two side spans. The sagging curvature of the side spans enables the vertical load on these spans to be carried primarily by a tension in the deck. The raking legs carry the compression from the centre span and the tension from the side spans. In addition, the legs flare as they reach the beach on the South Bank and the river wall on the North Bank to ensure stability.

The bridge was to be made from welded stainless steel. The legs were to be closed tubes with internal stiffeners. The deck was to be of similar construction with a U-shaped cross-section and inner and outer skins of stainless steel, again with internal stiffeners.

The bridge was to be made in sections and transported to site on barges where it was to be erected on temporary supports.

Outline Geometry

As with the road bridge study, the geometry of the footbridge is defined using the analytic function of a complex variable given in 4.1.1.3(1). The final bridge and its site location are illustrated in figures 4.1.1.3a and 4.1.1.3b.

A first map produced from an array of two sources and two sinks was used to generate the cross-sections of the bridge shown in figure 4.1.1.3c. Having produced this map, a second map arising out of the flow of a uniform stream over a row of vortices was produced using the relationship given in 4.1.1.3(5).

The two maps were then combined, by projecting the co-ordinates of the first map onto the function that produced the second map in order to create the 3-dimensional object, which embodies the characteristics of both the section and elevation maps. The details of the method used to generate both maps are discussed below.

Detailed Geometry - Bridge Sections

The computer program given in appendix B2.0, which incorporates the mathematical relationships given in appendix B2.3 produces the co-ordinates of the section geometry of the footbridge. This data is then read by the program in B2.1 and used to produce the section map shown in figure 4.1.1.3c.

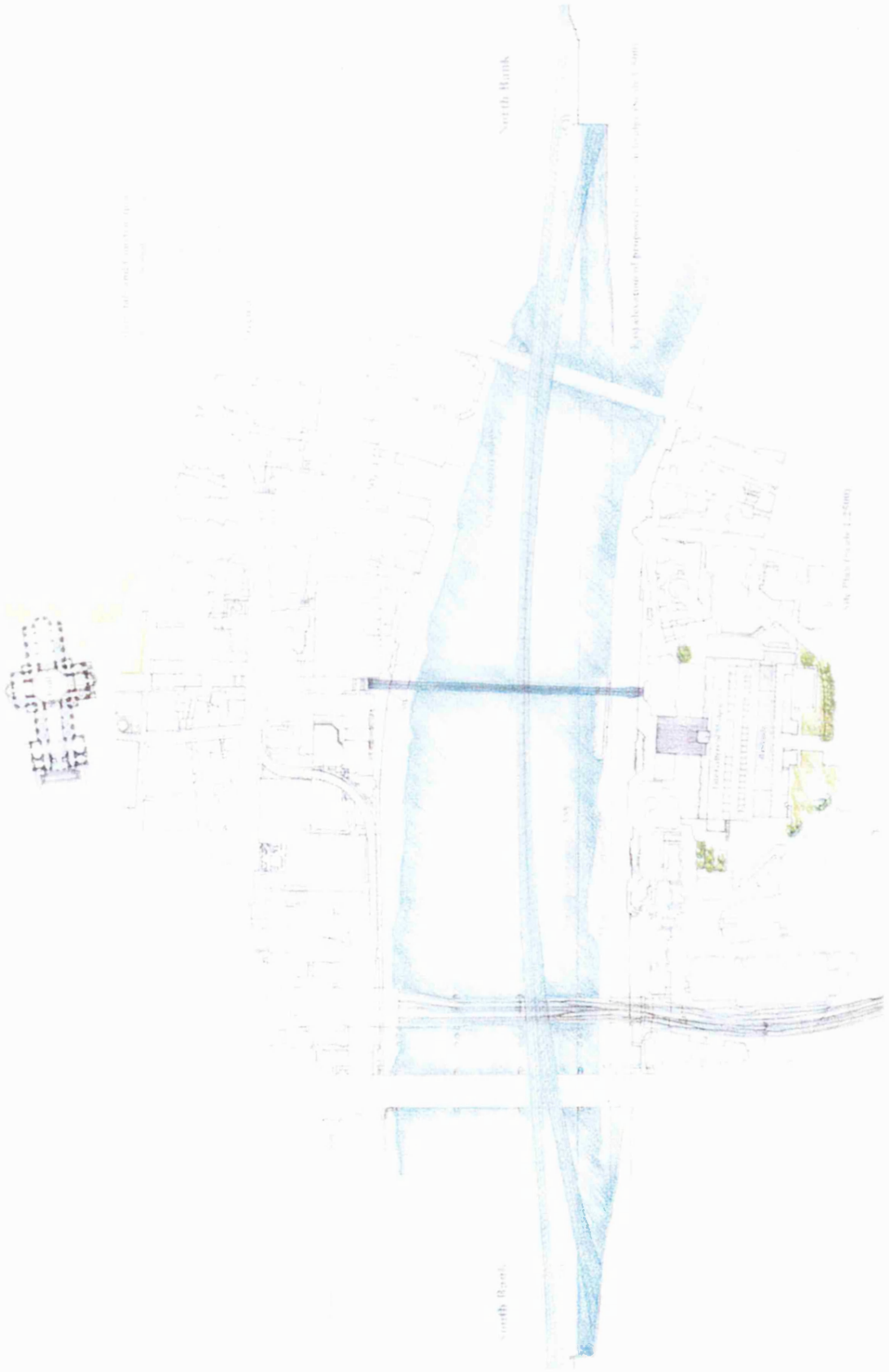


Figure 4.1.1.3a – Elevation and location plan of footbridge

Figure 4.1.1.3b Elevation of Footbridge study

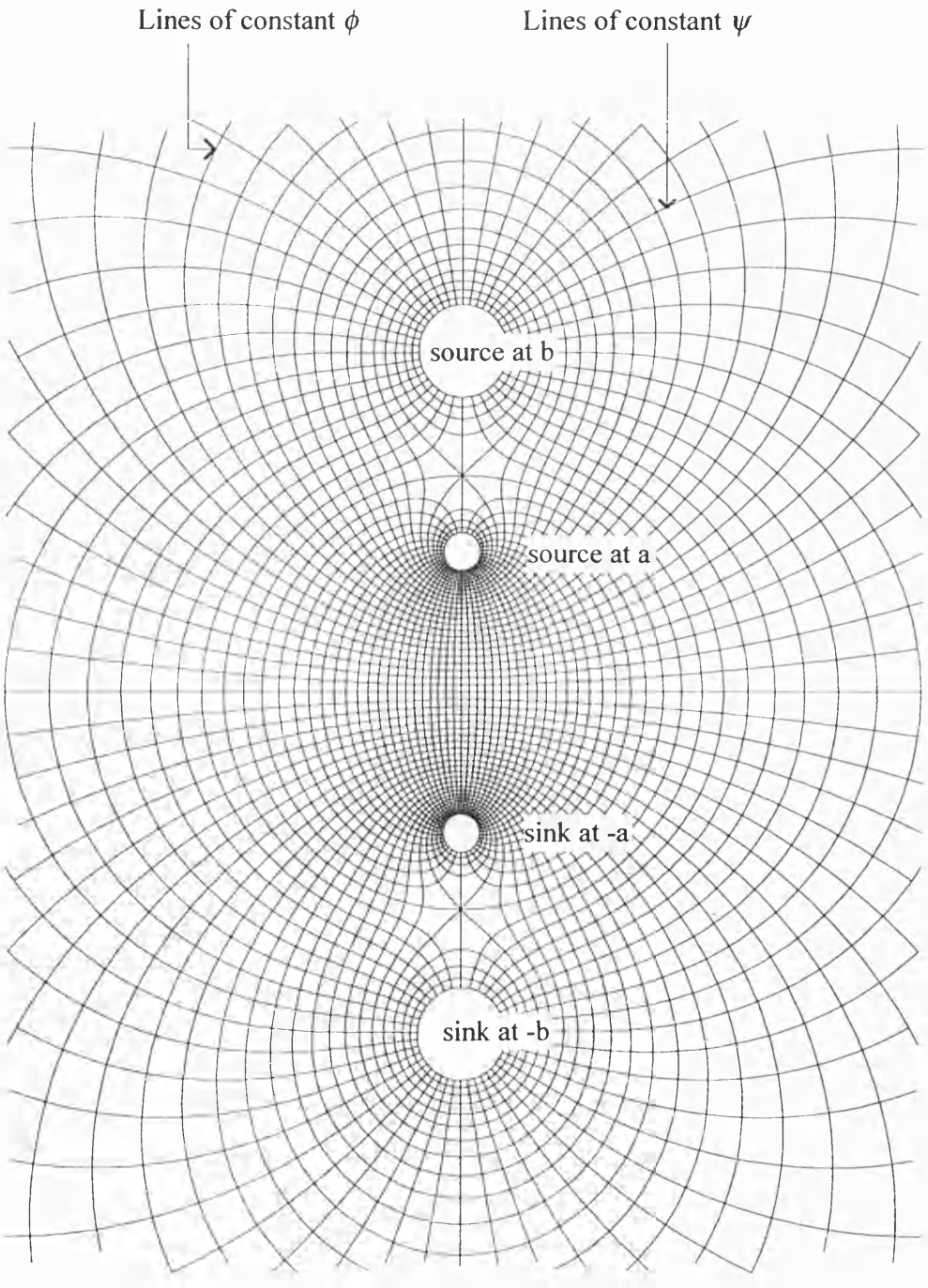


Figure 4.1.1.3c Extract of section map for Footbridge Study

The section map is generated from the interaction between two sources and two sinks given by the following relationship,

$$2\eta = A\log(z+ia)+A\log(z+ib)-A\log(z-ia)-A\log(z-ib). \qquad (1)$$

The sources are located at $z+ia=0$ and $z+ib=0$, and the sinks are located at $z-ia=0$ and $z-ib=0$.

The strengths of the sources are identical, so that an explicit expression for z can be produced from (1) using analytical methods. Details of the analytical method used in this case are given in appendix B2.3 and the resulting expression for z is given by,

$$z=-i\frac{(a+b)}{2}\left(\coth\eta\pm\sqrt{\coth^2\eta-\frac{4ab}{(a+b)^2}}\right), \qquad (2)$$

in which $\coth\eta$ is given by,

$$\coth\eta=\frac{\cosh\phi\sinh\phi-i\cos\phi\sin\phi}{\sinh^2\phi\cos^2\psi+\cosh^2\phi\sin^2\psi}. \qquad (3)$$

Details of the way in which the expression for $\coth\eta$ is derived are also given in appendix B2.3.

From this, initial co-ordinates of $z_1=x_1+iy_1$ corresponding to $\eta_1=\phi_1+i\psi_1$ are obtained. The map corresponding to this relationship is given in figure 4.1.1.3c where each equipotential line around the sources indicates a constant value of ϕ, and each curved radial line originating from the sources indicates a constant value of ψ.

Detailed Geometry - Bridge Elevation

The computer program given in appendix B2.0, which incorporates the mathematical relationships given in appendix B2.3 produces the co-ordinates of the elevation geometry of the footbridge. This data is again read by the program in B2.1 and used to produce the elevation map shown in figure 4.1.1.3d

This map is generated by the streamlines that arise from the flow of a uniform stream moving over a row of vortices. Any relationship, written in the form,

$$\eta = i \log z \ , \qquad (4)$$

indicates a vortex, rather than a source. The mapping function,

$$\eta = Vz - i\frac{au}{\pi}\log\left(\cos\frac{\pi z}{a}\right), \qquad (5)$$

where Vz represents the velocity of a uniform flow moving over an infinite row of vortices, describes the flow pattern shown in figure 4.1.1.3d. Appendix B2.3 gives details of how the function in (5) is arrived at using the generic form of a vortex flow given in 4.1.1.3(5) above.

The vortices are located at periodic intervals along the x axis, at a spacing of a, and u is a coefficient used to modulate the effect of the vortices. The number of sources are infinite because for values of $z = \ldots, -\frac{5a}{2}, -\frac{3a}{2}, -\frac{a}{2}, \frac{a}{2}, \frac{3a}{2}, \frac{5a}{2}, \ldots$, etc.,

$$\cos\frac{\pi z}{a} = 0 \text{ and } \log\left(\cos\frac{\pi z}{a}\right) = -\infty.$$

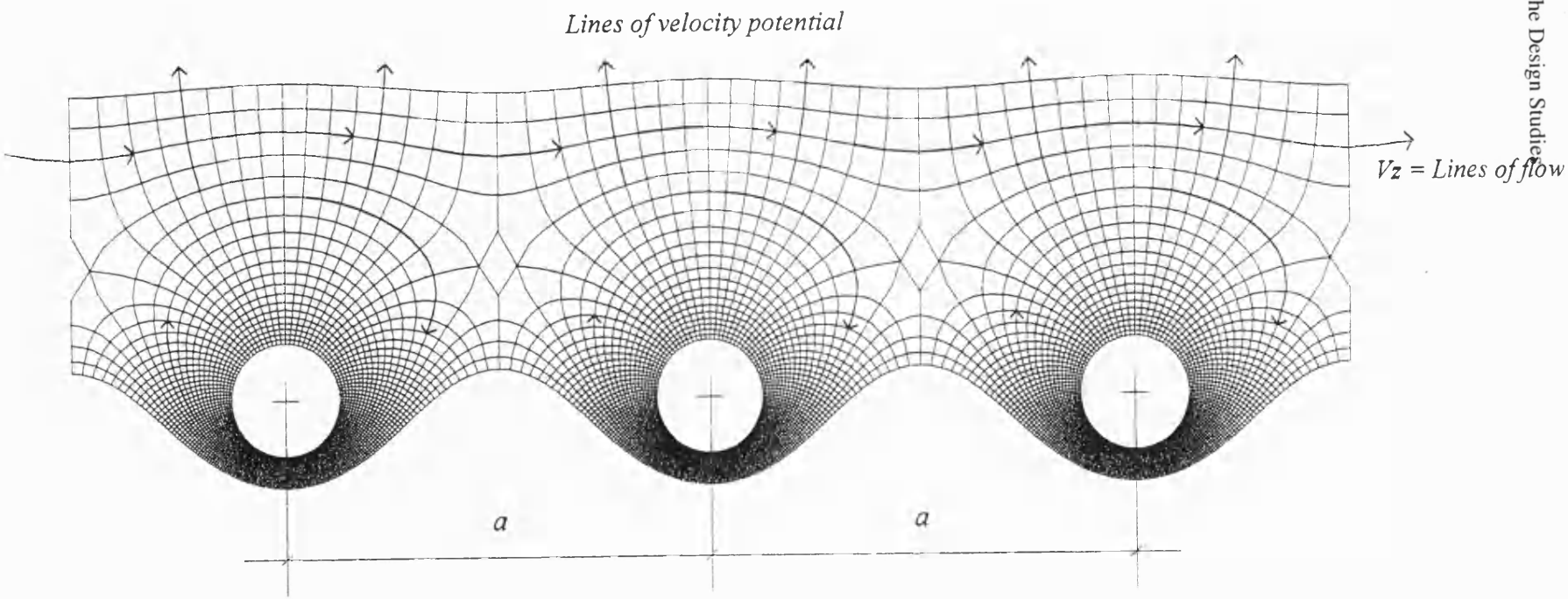


Figure 4.1.1.3d Conformal map used to generate Footbridge elevation

When referring to the computer program in appendix B2.0, it should be noted that the constant, *a,* is replaced by the symbol, *L,* and that the velocity, *V,* in 4.1.1.3(5) is replaced with $\alpha\frac{\pi}{L}$.

In figure 4.1.1.3d, the circumferential lines of clockwise rotational flow around each vortex are streamlines of constant ψ. The radial lines lying perpendicular to the streamlines are lines of constant velocity potential, ϕ.

The velocity is inversely proportional to the spacing between the lines, so that as the streamlines increase in their distance from the vortex origin, their velocity decreases. The streamlines closest to the origin of the vortex have the greatest velocity. At the position of the vortex or singularity, the velocity of the flow tends to infinity.

It is this combination of a continual increase in spacing between lines and a gradual flattening of the lines of flow due to a decrease in the effect of the vortex that accounts for the fan-like nature of the map. The fan-like configuration makes it possible to create the relative difference in curvature between the deck and supporting arch on the bridge elevation.

The line of greater curvature represents the supporting arch structure abutting the riverbanks, whilst the line of lesser curvature represents the deck surface, which is selected to comply with pedestrian gradient requirements.

Having obtained the required map geometry of the bridge cross-section, and having also obtained the required map geometry of the bridge elevation, the two functions are combined to produce the 3-dimensional object. This is achieved by projecting the co-ordinates of the cross-section onto the function that produces the elevation, so that,

$$z_2 = x_1$$

$$\psi_2 = y_1 \quad ,$$

where z_2 is the third co-ordinate appearing out of the plane of the elevation, set equal to x_1, and where ψ_2, is set equal to y_1. The third co-ordinate z_2 is not to be confused with the

complex number $z_2 = x_2 + iy_2$. The program given in B2.2 is used to produce the final detailed wire-mesh drawing of the bridge corresponding to this re-mapping.

The circumferential lines of constant ϕ_1 on the section map relate to the lines of constant velocity, ϕ_2 on the elevation map, but not in a direct manner. The relationship between them relies on ϕ_1 varying as a function of ϕ_2, in order for ϕ_1 to be mapped to the appropriate positions of ϕ_2, on the elevation map.

Mapping the section co-ordinates, x_1 to z_2, y_1 to ψ_2 and ϕ_1 to some function of ϕ_2 using the relationship given in (2) used to generate the elevation map, produces new co-ordinates, x_2 and y_2. Newton's method is used to solve the expression in (2), since the resulting polynomial cannot be solved numerically to give explicit values of x_2 and y_2 as functions of ϕ_2 and ψ_2.

In general, the advantage of a mathematical generator lies in the fact that the intensity of curvature along any row of lines on the elevation can easily be adjusted by modifying the mathematical function very slightly. It was therefore possible to adjust the bridge by going through a process of trial and error, until its profile was contained entirely within the geometric constraints of the brief.

4.1.1.4 The Wall Studies

The wall studies were developed as part of an entry in a design competition to create a museum and memorial for Senegal. The museum building was to be sited on the Cap Vert peninsula, which is located on the western most tip of the African continent. The memorial was to be sited on the Island of Gorée, in the attempt to reinforce the symbolic nature of its role in the quest for universal liberation from slavery.

Given the cultural significance of the project and the need to take into account, the particular climatic factors of Senegal, the enclosing wall was deemed an element deserving close study. The aim was to develop a wall module that would satisfy climatic demands for good thermal insulation and shading, so as to protect the interior activities against the extremes of heat and sunlight. It was also intended that the visual character of the wall would have strong associations with the richness of North African architectural heritage.

Initial inspiration for the wall was derived from skeletal remains of radiolarian shells, images of which are shown below in figure 4.1.1.4a.

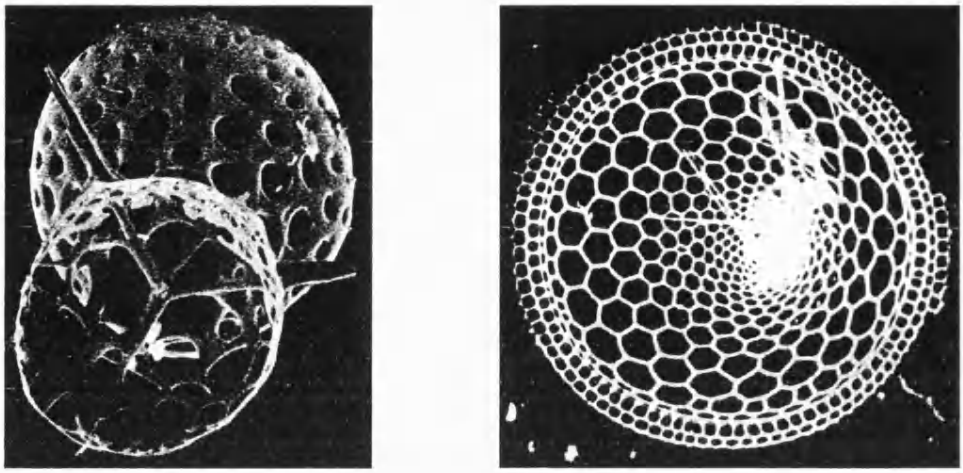


Figure 4.1.1.4a Radiolarian skeletons

The wall studies incorporate experience drawn from previous studies on lines of flow generated by the interaction between sources and sinks. In order to create a wall panel of large enough dimensions, it was decided to arrange a number of alternate rows of sources and sinks. Variations of the resulting wall panel could be used as building blocks for the façade.

The result of an arrangement of alternating rows of sources and sinks is shown in figure 4.1.1.4b. The analytic function developed for it corresponds to,

$$\eta = \sum_{n=-\infty}^{\infty} \log\left(\sin\frac{\pi(z+2nib)}{a}\right) - \sum_{n=-\infty}^{\infty} \log\left(\sin\frac{\pi\left(z+\frac{1}{2}a+(2n+1)ib\right)}{a}\right). \qquad (1)$$

The sources are arranged at periodic intervals of a along the horizontal axis, and at even intervals of $2nb$ along the vertical axis. The sinks are arranged at alternating intervals of $\frac{1}{2}a$ along the horizontal axis, and at odd intervals of $(2n + 1)b$ along the vertical axis.

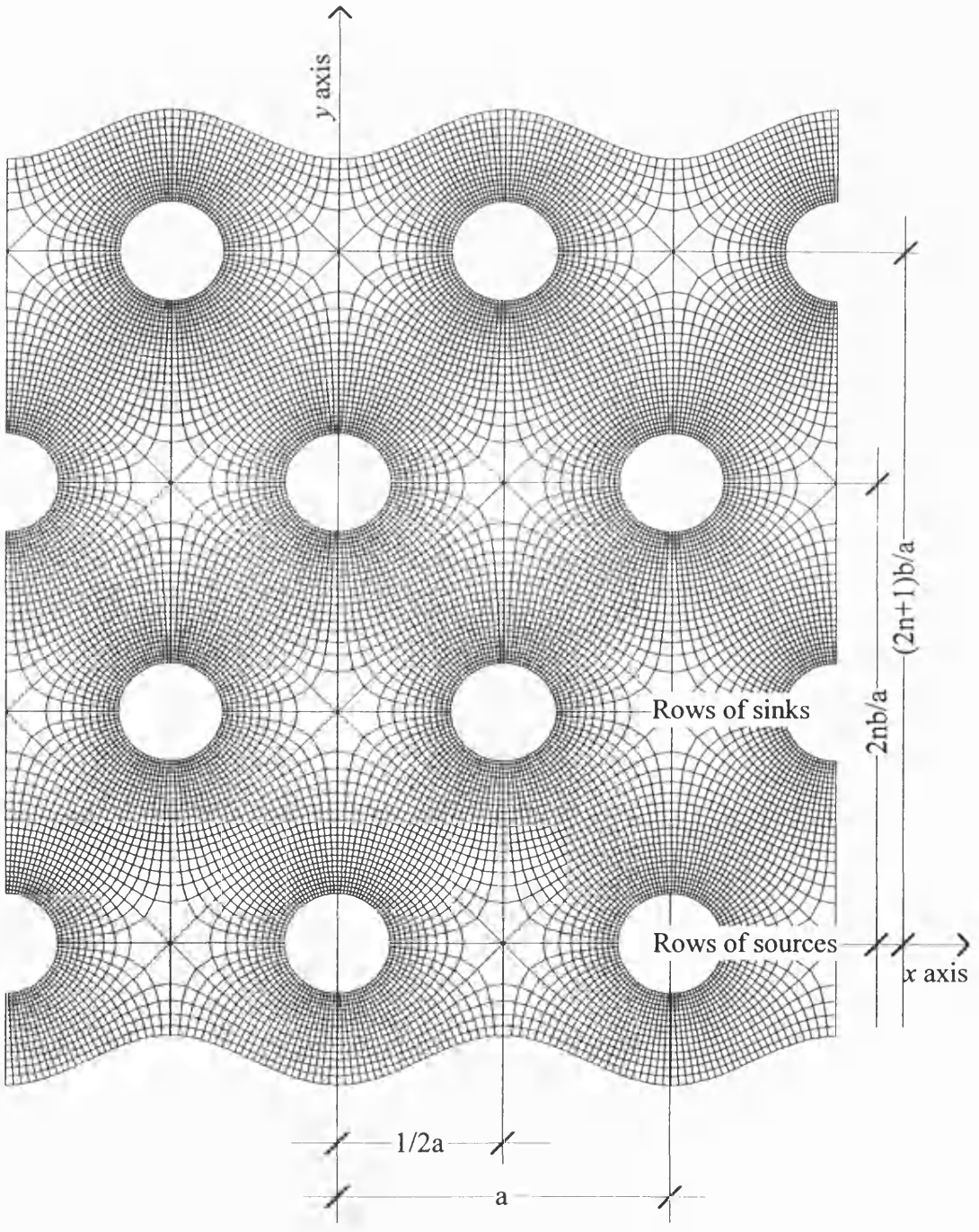


Figure 4.1.1.4b Extract of map for Wall Studies

The positive $\sum$ indicates the rows of sources, whilst the negative $\sum$ indicates the rows of sinks. The expression in (1) can be rewritten to give

$$\eta = \sum_{n=-\infty}^{\infty} \log\left(\sin\frac{\pi(z+2nib)}{a}\right) - \sum_{n=-\infty}^{\infty} \log\left(\cos\frac{\pi(z+(2n+1)ib)}{a}\right) \qquad (2)$$

which creates an identical map, because the substitution of the sine in (1) with the cosine in (2) effectively shifts the relative positions of the sinks by $\frac{a}{2}$, hence the omission of $\frac{a}{2}$ in (2).

The method to find the co-ordinates of z uses Newton's method given in 4.1.1.2(11), since again in this case, it is not possible to produce an explicit expression for z in terms of η using the 'elementary' functions available in a computer program. In the computer program the infinite sums are replaced by finite sums with a large number of terms.

In order to convert the map in figure 4.1.1.4b into 3-dimensions, it was simply necessary to choose an appropriate function of ϕ for the third co-ordinate. In addition, the wall is wrapped around to enclose space.

The result is shown in figure 4.1.1.4c and the program used to generate it is given in appendix D3.0. A 3-dimensional rendered view of the wall is also shown in figure 4.1.1.4e.

As part of the process to develop the wall panel, an important issue was that it should not appear repetitive since the scale of the proposed museum called for a certain degree of diversity, which it was felt, should be reflected in the façade.

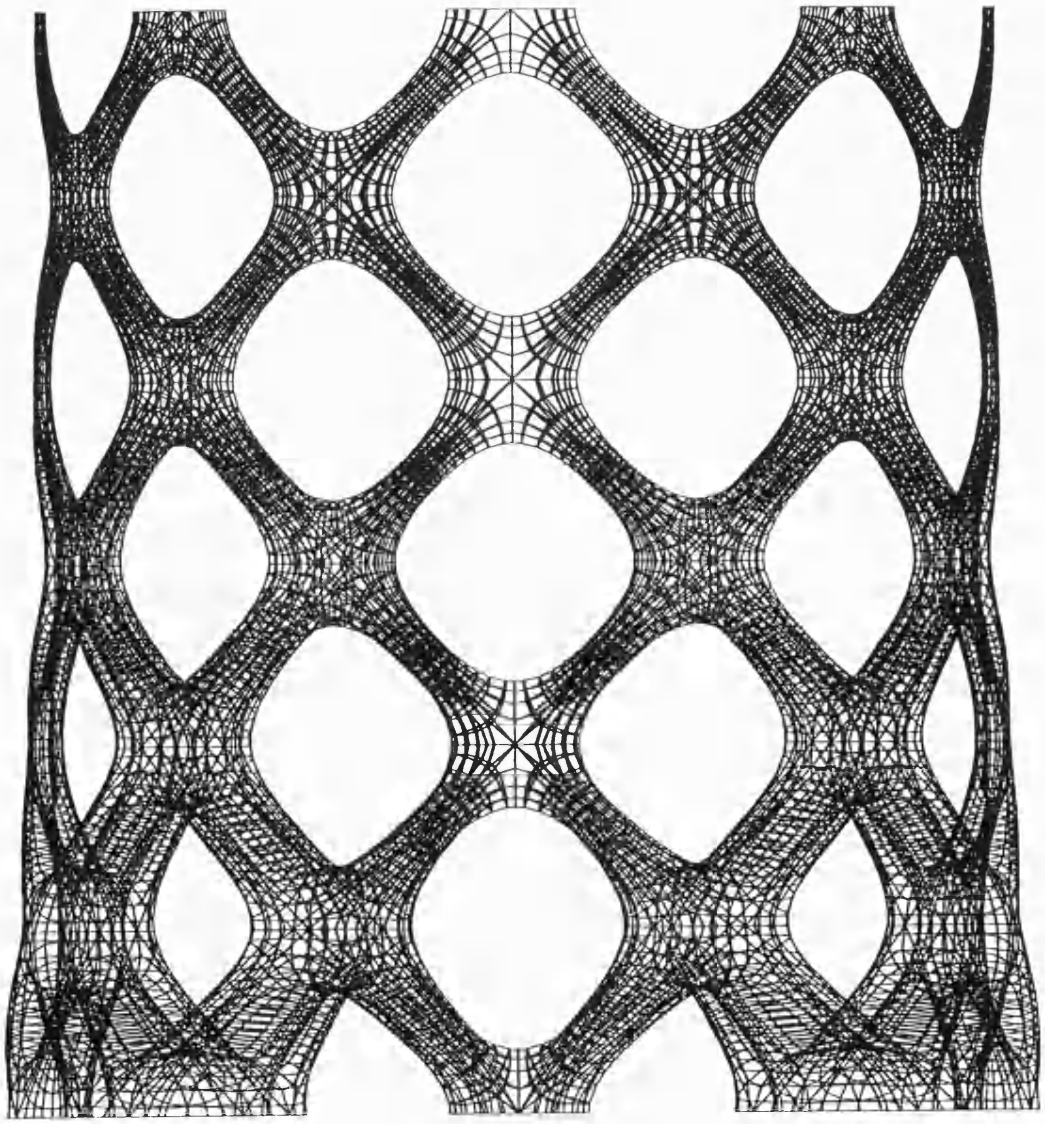


Figure 4.1.1.4c Elevation of wall study

In order to achieve this, a randomness factor was incorporated into a typical wall module to vary the distribution and size of its openings. This feature also meant that it was more consistent with the nature of radiolarian shells. Figure 4.1.1.4d shows the result of the application of a random order onto a typical module.

Another outcome of the random order was an increased level of flexibility in planning the spaces behind the façade. Large openings spanning a double height space could be used for public areas, whilst small openings reflected spaces requiring greater privacy.

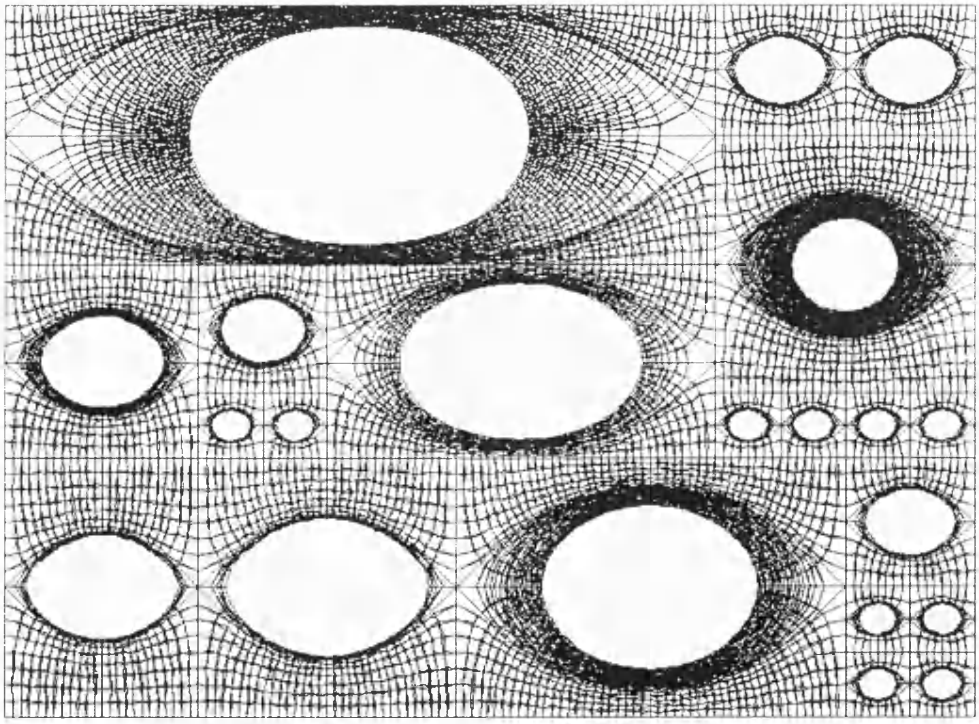


Figure 4.1.1.4d Study for random wall

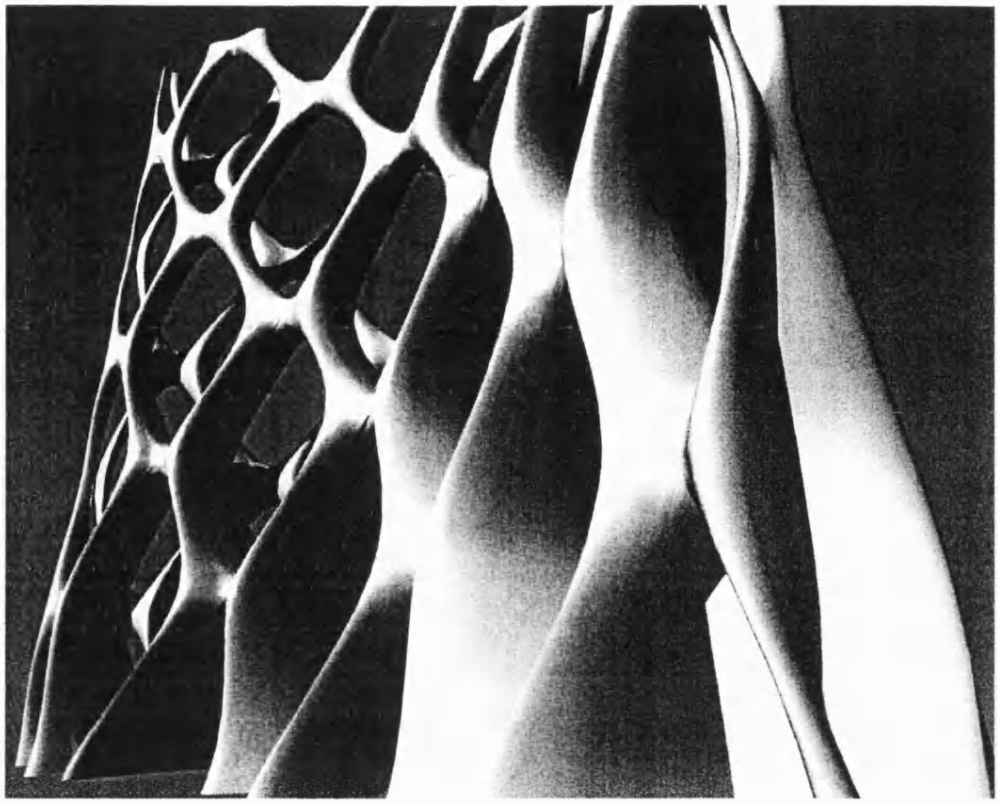


Figure 4.1.1.4e 3-dimensional view of wall study

4.1.1.5 The Sculpture

The starting point for the sculpture shown in figure 4.1.1.5a was again the pattern of alternating rows of sources and sinks shown in figure 4.1.1.4b. The initial co-ordinates are curtailed and distorted using the complex function,

$$\begin{aligned} x + iy &= i\cos(\xi + i\varsigma) \\ &= i\left[\cos\xi\cos(i\varsigma) - i\sin\xi\sin(i\varsigma)\right], \\ &= \sin\xi\sinh\varsigma + i\cos\xi\cosh\varsigma \end{aligned} \qquad (1)$$

which produces the pattern shown in figure 4.1.1.5b, where x and y are the Cartesian co-ordinates of the new map, which result from applying the mapping function in (1) to ζ and ς, which represent the co-ordinates of the previous map given in figure 4.1.1.4b. The shape in 4.1.1.5a is produced first by mapping x and y onto an ellipsoid and then twisting the result.

The ellipsoid function corresponds to

$$r = \sqrt{\left[\sin 4\pi \frac{\frac{y-a}{2}}{4a} \cosh 4\pi \frac{\frac{a}{2}+\frac{a}{4}}{4a}\right]^2 \left[\cos 4\pi \frac{\frac{y-a}{2}}{4a} \sinh 4\pi \frac{\frac{a}{2}+\frac{a}{4}}{4a}\right]^2}.$$

Further details with regard to twisting the ellipsoid are given in appendix D4.0, which contains the full program to generate the sculpture.

The wall studies demonstrate the versatility offered by the maps produced by sources and sinks, in that they can be adapted to several architectural and artistic applications.

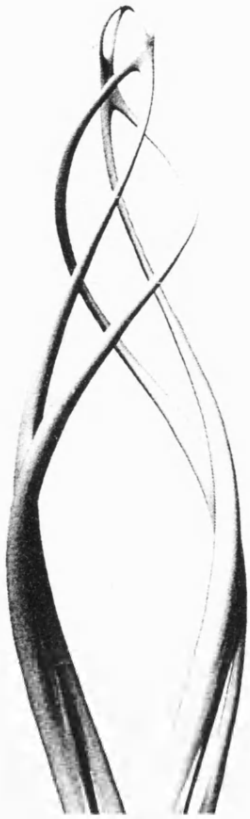


Figure 4.1.1.5a The Spiral sculpture

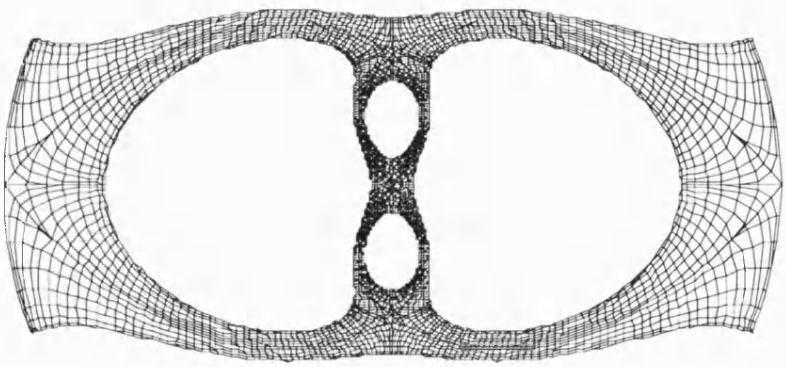


Figure 4.1.1.5b Plan of the map for the spiral sculpture prior to twisting

4.2 The Millennium Dome Rest Zone

Introduction

The appointment of the Richard Rogers Partnership to develop a design for the Rest Zone follows a previous involvement by the artist, Anish Kapoor. Kapoor's curvilinear sculptures have won him high acclaim due to their unique ability to invert space. Kapoor creates the illusion of depth through the use of voids and his objects are soaked in the rich intense monochromes of indigos, blood reds and flawless whites.

The Rest Zone by RRP is very similar in appearance to Kapoor's 1995 piece, *White Dark IV* made from wood, fibreglass and paint, shown in figure 4.2a. It resembles a giant kidney bean.

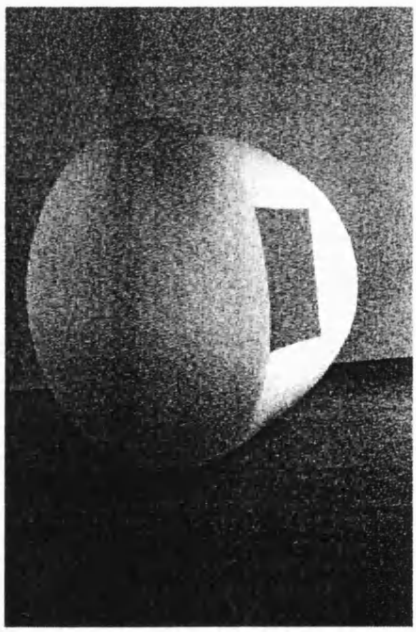


Figure 4.2a Kapoor's *White Dark IV, 1995* (Collection Artimo Foundation)

Like the Body, Mind, Work and Local Zones, the Rest Zone is one of the event spaces incorporated into the 300m diameter Dome enclosure. The object of the Rest Zone is to provide a place of tranquillity for secular retreat. This quest to capture space that encourages silent contemplation draws on the very essence of Kapoor's ephemeral art. Most of Kapoor's pieces create the impression of infinite depth by using curvilinear surfaces to transport the viewer gradually into a *distant* void.

RRP's proposed design for the Rest Zone was a modified torus, which defied precise and detailed description using conventional computer drawing techniques. As a piece of sculpture, it would have been realised using the intensive maquette and hand-crafting procedures used by Kapoor and the skilled metalworkers, plasterers and carpenters whom he often engages to assist in executing his works. As a piece of architecture however, it required more accurate geometrical definition.

The information describing its geometry was passed to a boat builder who uses IGES (Initial Graphics Exchange Specification), an optional output file available in most CAD packages, as input for the driver of his cutting tool.

Buro Happold, the consulting engineers for the project approached Chris Williams to assist the Richard Rogers Partnership with the geometry-finding process. It was suggested that the process of developing the form might provide a useful case study in this dissertation to explore the application of mathematical methodology to shape finding in design.

The author's participation in the geometry-finding process involved evaluating changes required by the architects and assisting with their incorporation into the program. This provided the author with first-hand experience and a good knowledge of the computer program as well as with its versatility and speed at effecting major changes.

The aim of the following discussion will be to record the process of developing the form of the Rest Zone and to explain the mathematical language used to generate it such that the experience of this process will permit similar techniques to be applied to future projects by architects and engineers.

Design Objectives

An important requirement for the Rest Zone was, that when viewed from inside the dome, it appeared to be an object associated with the dome space but that once inside the Rest Zone, it appeared to be an object in its own right, totally isolated from its immediate environment.

The definition of its geometry involved describing two surfaces - first, the external skin with its supporting framework of structural ribs and second, the internal skin which

created the effect of a cocoon, being pierced only by small openings to form the entrances and exits.

The form of the Rest Zone is 'doughnut' shaped with a hole to allow an elevated external bridge to pass through the building while remaining separate from the inside. The lower third of the doughnut is cut off by the floor surface and therefore does not exist in the real building.

The purpose of the next section will be to describe three aspects of the geometry-finding process of the Rest Zone – firstly, its outline geometry, secondly, its detailed geometry and thirdly, in brief, the process of its deformation.

4.2.1 Rest Zone Geometry-finding Process

Outline Torus Geometry

In broad terms, the Rest Zone is generated from a torus, which is then deformed. The outline geometry establishes the schematic geometry, which is obtained by rotating a circle of radius r around another circle of radius R.

In order for a hole to exist in the middle of the torus, R has to be bigger than r. As the small circle travels through angles of θ from zero to 360 degrees along a path circumscribed by the large circle, co-ordinates for all points along the perimeter of the minor circle are derived using trigonometric methods.

The Mathematical Relationships

Consider a minor circle of radius r, which rotates around a major circle of radius R. At an angle θ, marking the position of the minor circle relative to the horizontal plane H, the x, y and z co-ordinates of a certain point P on the minor circle that forms an angle ϕ relative to the x - z plane are given by the trigonometric relationships given in 4.2.1(1), and illustrated in figures 4.2.1a, b and c.

$$x = \left(R + r\cos\phi\right)\cos\theta$$

$$y = r\sin\phi \qquad (1)$$

$$z = (R + r\cos\phi)\sin\theta.$$

From the relationships in (1), the x, y and z co-ordinates of points on the minor circle for values of ϕ between zero and 360 degrees are obtained. Varying θ gives the entire set of co-ordinates of the minor circle as it travels around the large circle.

Appendix C1.1 shows these relationships incorporated into a computer program written in C++. The result is a *dxf* file, which can be read into most CAD packages. Figure 4.2.1c was plotted from this dxf file.

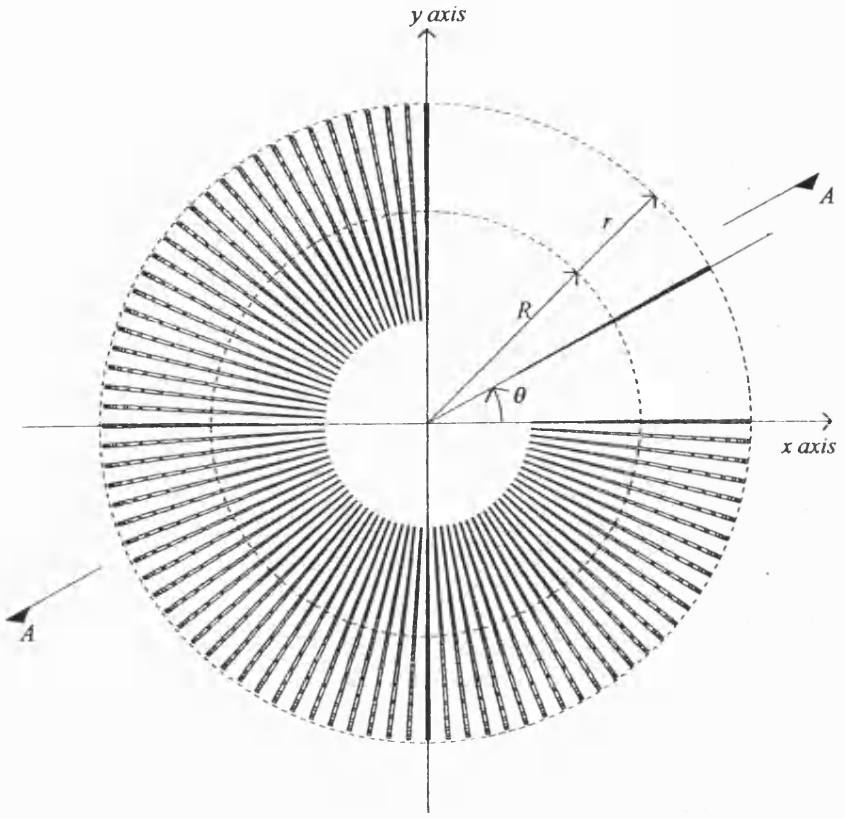


Figure 4.2.1a – Torus elevation showing mathematical relationships

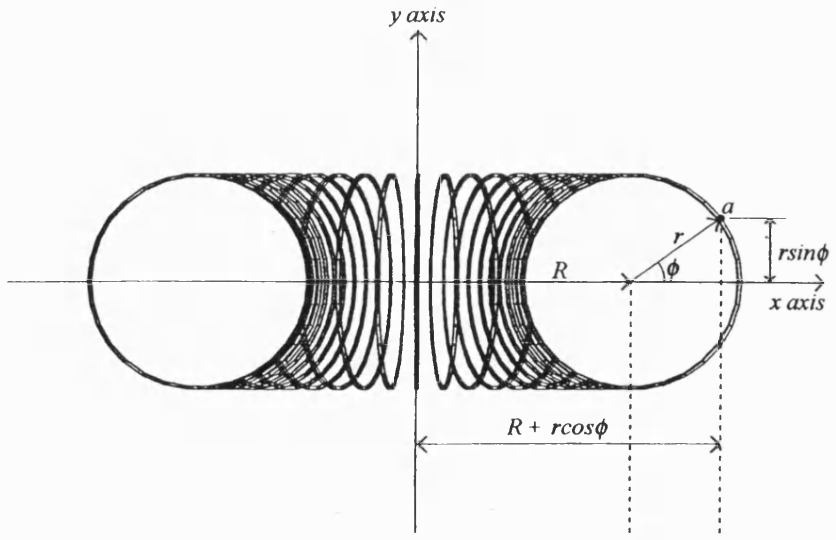
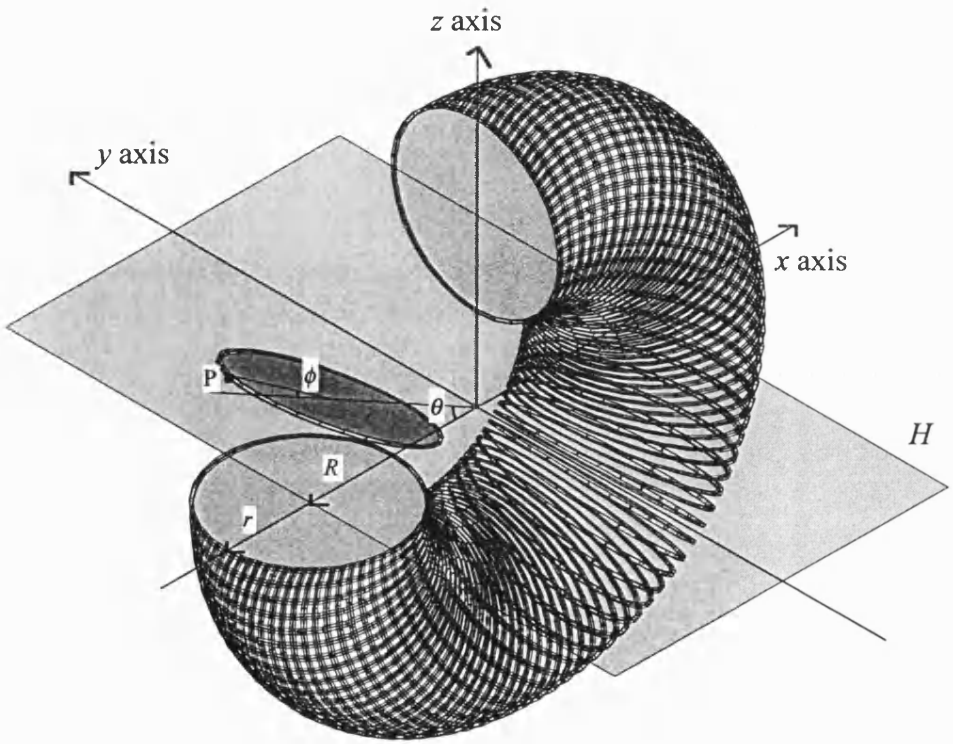


Figure 4.2.1b – Plan A - A of torus



$$x = (R + r\cos\phi)\cos\theta$$

$$y = r\sin\phi$$

$$z = (R + r\cos\phi)\sin\theta$$

Figure 4.2.1c 3-d view of torus showing the mathematical relationships

Detailed Torus Geometry - General

This section discusses the most important aspect of the geometry-finding process, which involves the description of the torus surface by means of parallel cross-sections. This approach was adopted in order to maintain consistency between the surface tiling pattern on the torus and RRP's preferred layout for structural ribs. Parallel cross-sections also made it easier to incorporate the form onto general arrangement drawings, and enabled data for manufacturing purposes to be extracted more readily.

The concept of intersecting the surface of a torus with vertical planes is not new. In 150BC Perseus of Ancient Greece had considered the idea of intersecting planes, which lay parallel to the axes of symmetry of tori. Perseus found that at a certain point on the torus, the vertical plane intersecting it created a figure of eight, or *hippopede,* as Eudoxus called it. According to Needham (1997), the Ancient Greeks called these figure of eight curves, *spiric sections of Perseus.*

In more recent times, figure of eight curves have been named *Cassinian curves,* after Giovanni Cassini (1625 -1712), who applied them in his endeavours to describe planetary motion. Cassini's studies of figure of eight curves followed Newton's work on elliptic orbits in *Principia,* and the curves later had applications for James Bernoulli in understanding the behaviour of elliptic functions.

The basic torus constructed from vertical slices is illustrated in figure 4.2.1d and the final design is shown in figure 4.2.1e. The process involved in presenting it in this form is rather like building a physical model where the peculiarities of its shape only come to light through a three-dimensional study of the problem. In this case, the three-dimensional problem is given a mathematical interpretation.

Construction of Vertical Slices

The mathematical problem of presenting the undeformed torus using a series of parallel cross-sections involves determining the co-ordinates x, y and z and the angles ϕ and θ that correspond to each point on a slice. The equations describing the surface of the torus given in 4.2.1(1) are not abandoned; however, new parameters are simply added to enable

the torus to be represented differently, whilst remaining constrained to move along the existing torus surface.

To assist in locating the position of points on the torus surface, it is customary to express the Cartesian co-ordinates of the surface as functions of a system of surface co-ordinates, u and v. In this case, it is convenient to label the points on the surface using the integers i and j to replace u and v, in order to maintain consistency with the computer program where the finite number of points are labelled by integers.

The vertical lines are denoted by i, whilst the opposing curved lines are denoted by j. The value of x depends only upon i and that the value of ϕ, depends on i and j. If x and ϕ are known, y and z can be readily calculated.

The limits of x are easy to determine, since they relate to the physical dimensions of the torus and the spacing between the ribs. The method to obtain values of x, is given in the next section on 'Determining the rib intervals'.

In the case of ϕ, we can suppose that in general, its limits lie between 0 and 180° for the half of the torus on which y is positive or zero, except beyond the centre hole where its limits are not constant. The section entitled 'Limits of ϕ, shows how values of ϕ are determined.

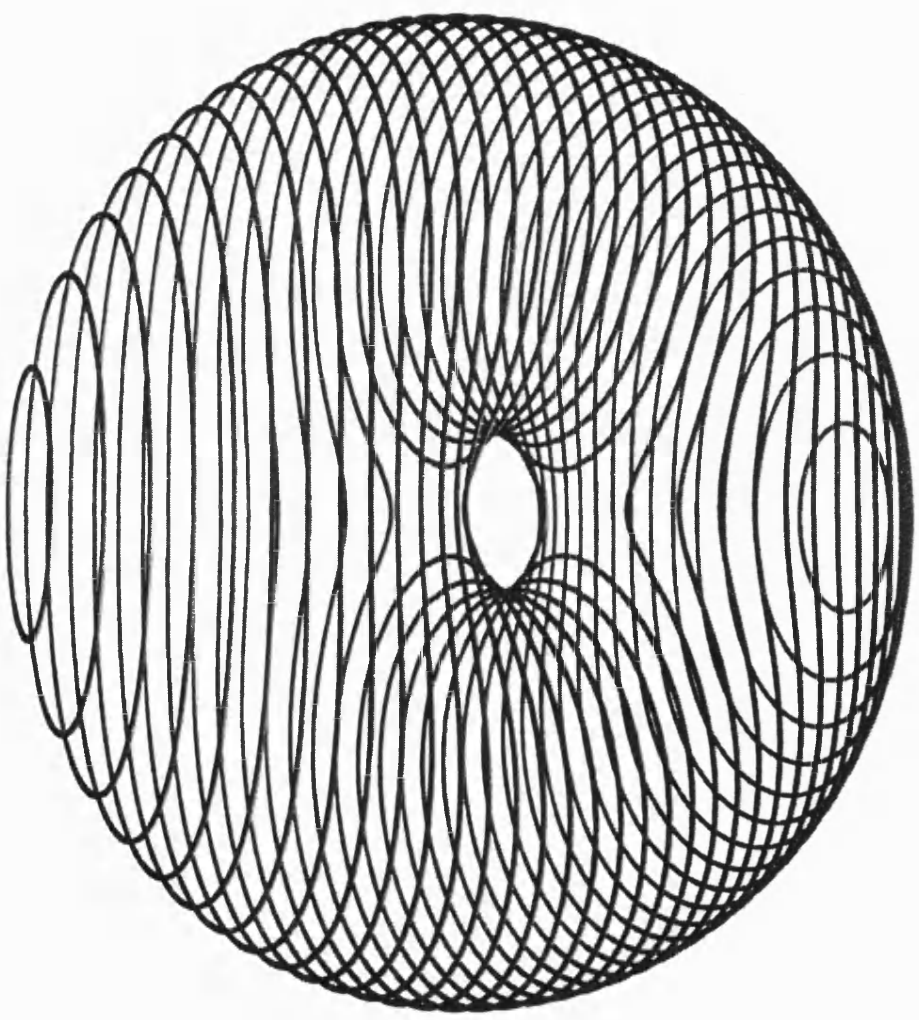


Figure 4.2.1d Undeformed torus constructed from vertical slices

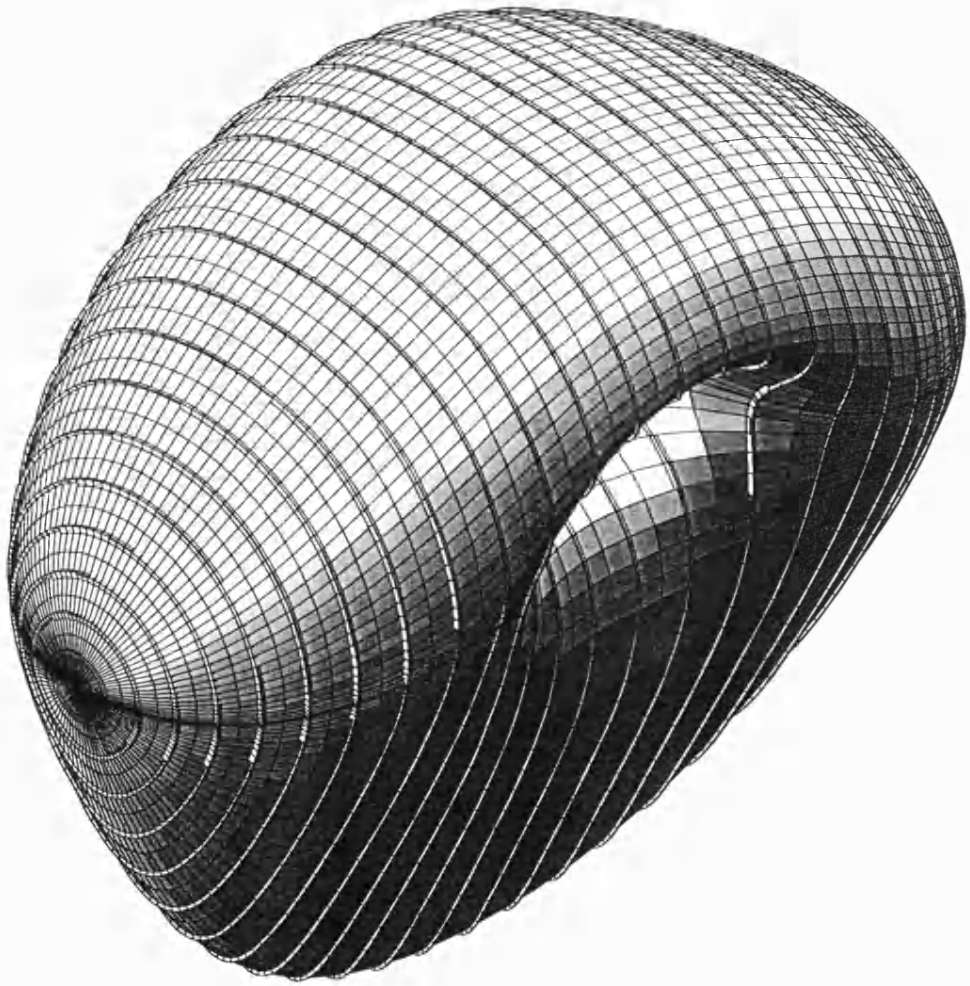


Figure 4.2.1e Isometric view of Rest Zone

Determining the rib intervals

The intervals between structural ribs are obtained by using a sine function, because RRP had a requirement for the intervals between ribs to diminish progressively towards the outer edges of the torus.

The method is illustrated in figure 4.2.1f, which shows that the diminishing interval is inversely proportional to the rib number, so that the interval for rib number 4, for example, is larger than that for rib number 26.

The distance from the centre of the torus to the inside edge of the hole, marked thus *, in figure 4.2.1f, is given by

$$R-r=\frac{31}{2}\sin\left(\frac{4}{26}.\frac{\pi}{2}\right),$$

where $\frac{31}{2}$ represents half the actual length of the torus, $\frac{4}{26}$ refers to the rib number positioned at the hole, expressed as a ratio of the total number of ribs. The distance to the outer edge of the torus is given by,

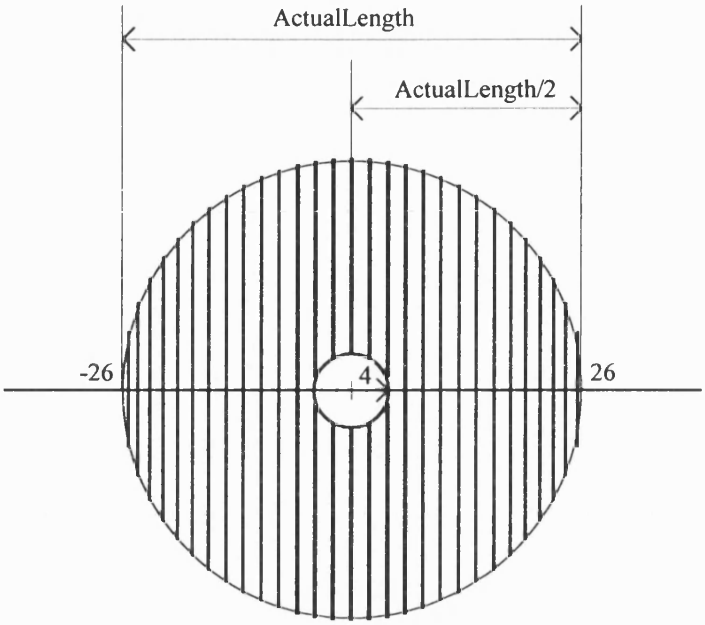
$$R+r=\frac{31}{2}\sin\left(\frac{26}{26}.\frac{\pi}{2}\right)=\frac{31}{2},$$

therefore the distance marked thus, **, in figure 4.2.1f is given by,

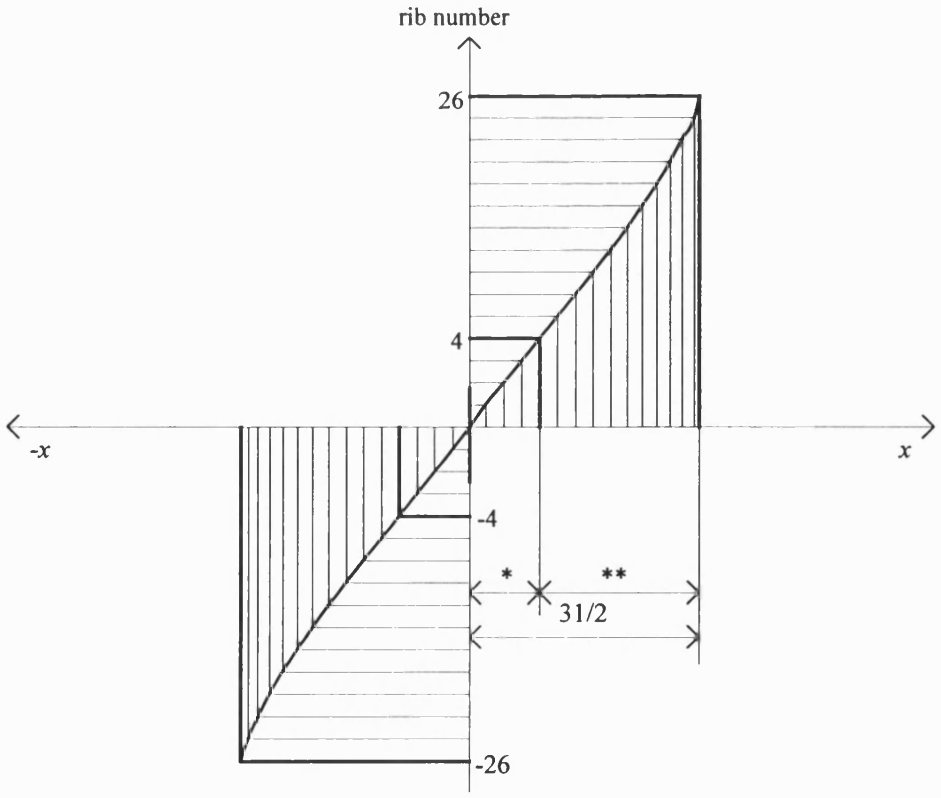
$$\left.\begin{aligned} 2r&=\frac{31}{2}\left(1-\sin\left(\frac{4}{26}.\frac{\pi}{2}\right)\right).\\ &\text{Also}\\ 2R&=\frac{31}{2}\left(1+\sin\left(\frac{4}{26}.\frac{\pi}{2}\right)\right).\end{aligned}\right\}. \qquad (2)$$

The value of x on the ith rib is

$$x=\frac{31}{2}\sin\left(\frac{i}{26}.\frac{\pi}{2}\right). \qquad (3)$$



Elevation of torus showing diminishing rib intervals



Graph of rib number as a sine function of *x*

Figure 4.2.1f Determining the rib intervals

Limits of ϕ

Along the apical points of the torus, the value of ϕ is 0 degrees, and around the hole at the centre of the torus, the value of ϕ is 180 degrees as shown in figures 4.2.1g – 4.2.1j. Figures 4.2.1k – 4.2.1m show the equivalent diagram in the deformed torus case. All vertical slices taken in the region of the hole create two closed curves on the upper and lower sectors of the torus.

At the position of the Cassinian slice, shown in figure 4.2.1g, the curves on the upper and lower sectors cross at their inner tips and again, the minimum and maximum values of ϕ are 0° and 180° on the half of the torus on which $y \geq 0$. In the region beyond the Cassinian slice, however, the maximum value of ϕ is less than 180°.

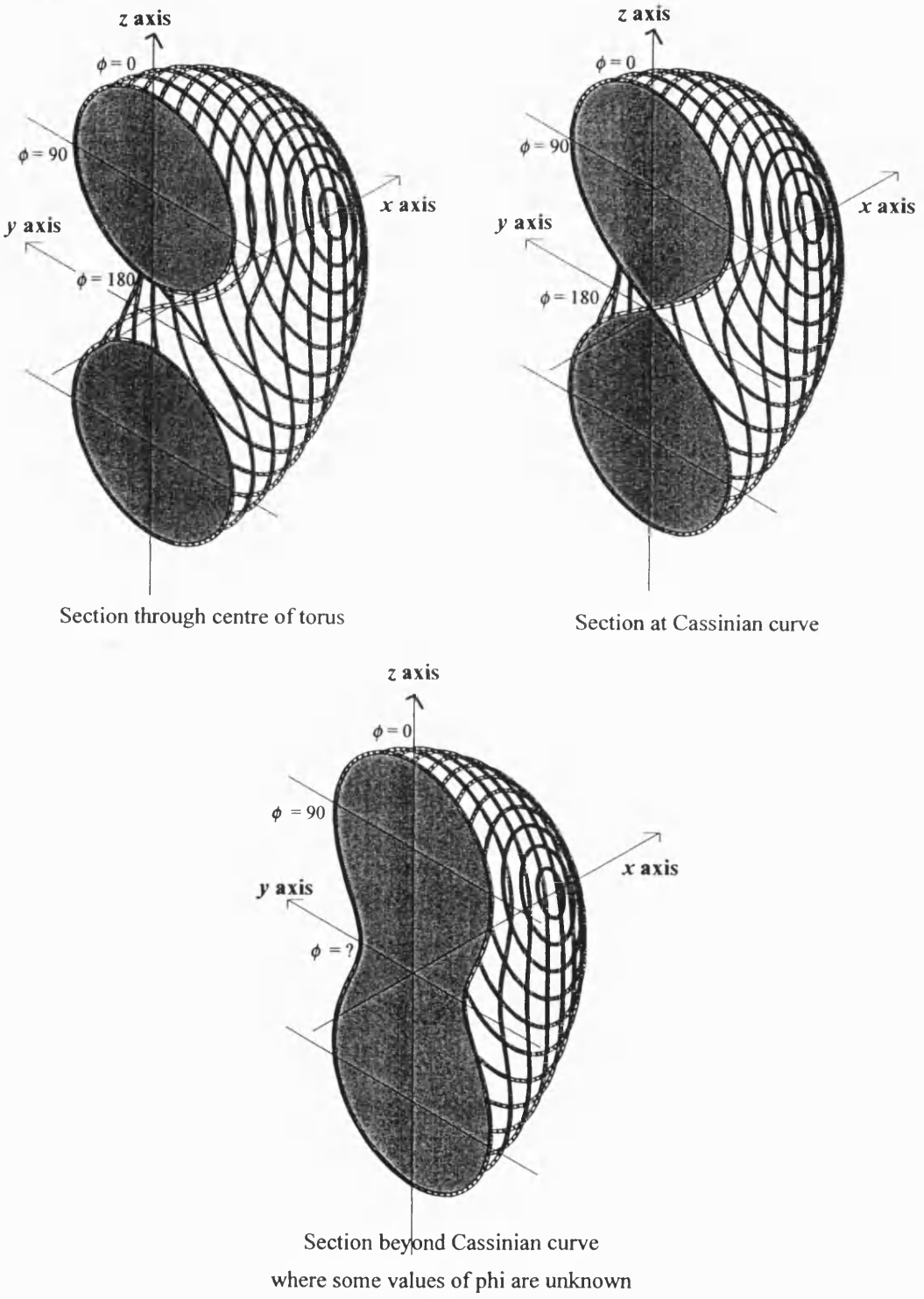


Figure 4.2.1g Vertical slices through torus

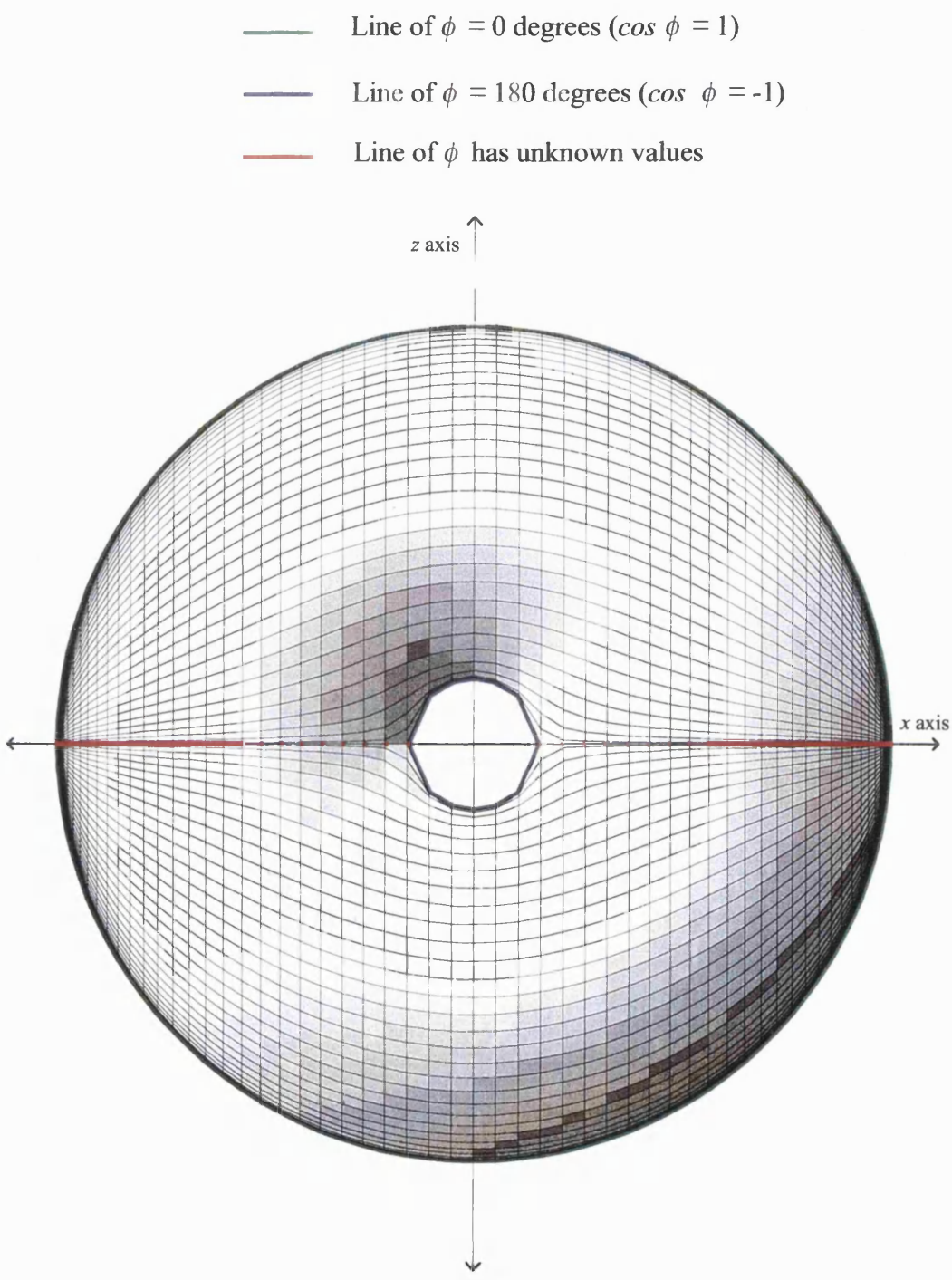


Figure 4.2.1h Front view of undeformed torus showing limits of ϕ

Line of ϕ = 0 degrees ($cos\ \phi$ = 1)

Line of ϕ = 180 degrees ($cos\ \phi$ = -1)

Line of ϕ has unknown values

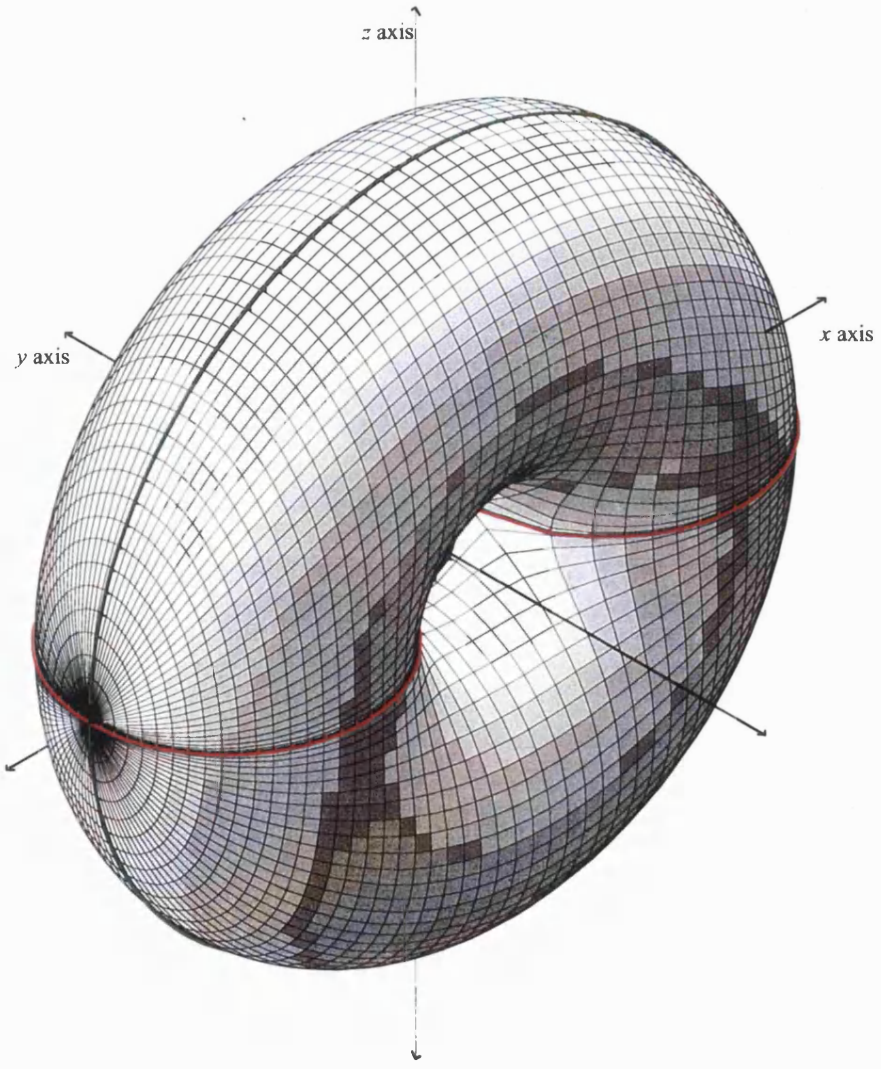


Figure 4.2.1i Axonometric view of undeformed torus showing limits of ϕ

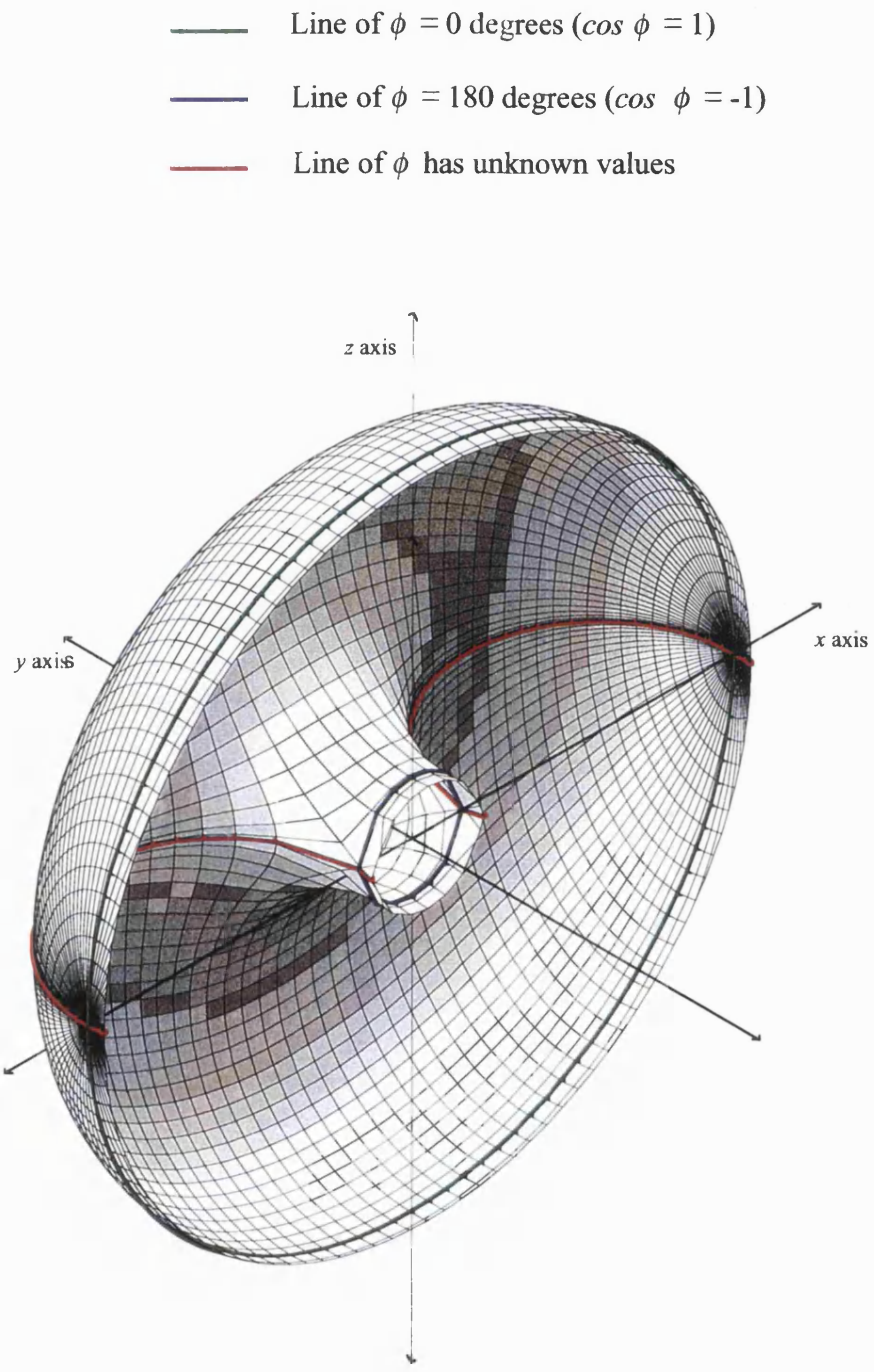


Figure 4.2.1j Cut-away axonometric view of undeformed torus showing limits of ϕ

Line of ϕ = 0 degrees (cos ϕ = 1)

Line of ϕ = 180 degrees (cos ϕ = -1)

Line of ϕ has unknown

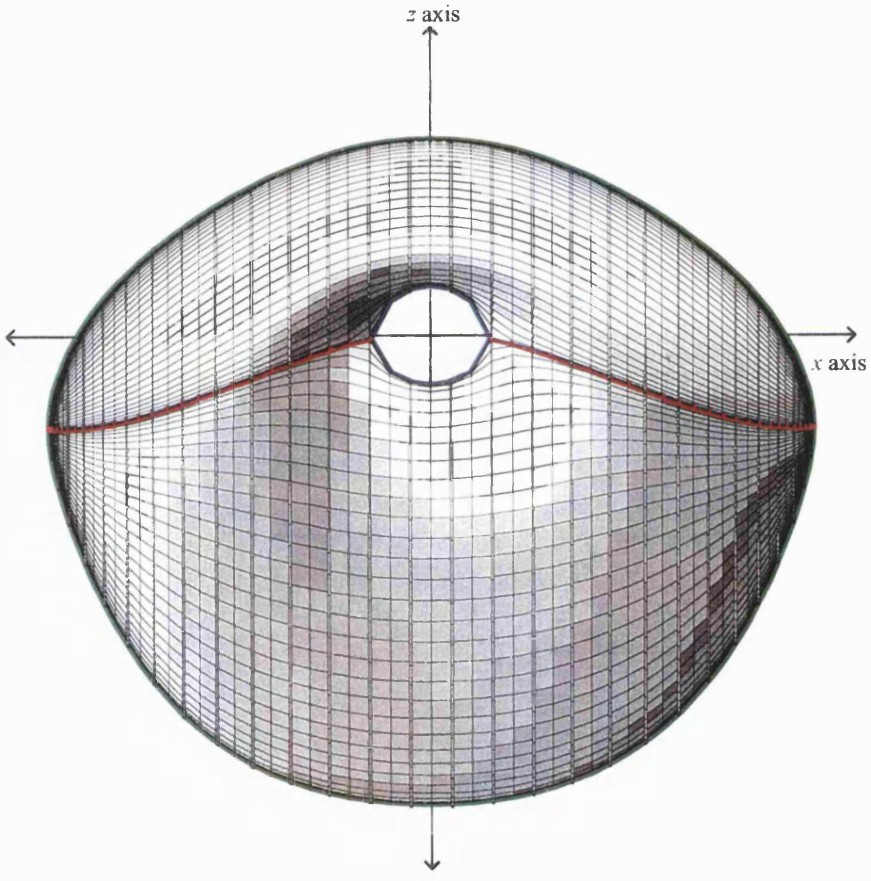


Figure 4.2.1k Front view of deformed torus showing limits of ϕ

Line of ϕ = 0 degrees (cos ϕ = 1)

Line of ϕ = 180 degrees (cos ϕ = -1)

Line of ϕ has unknown

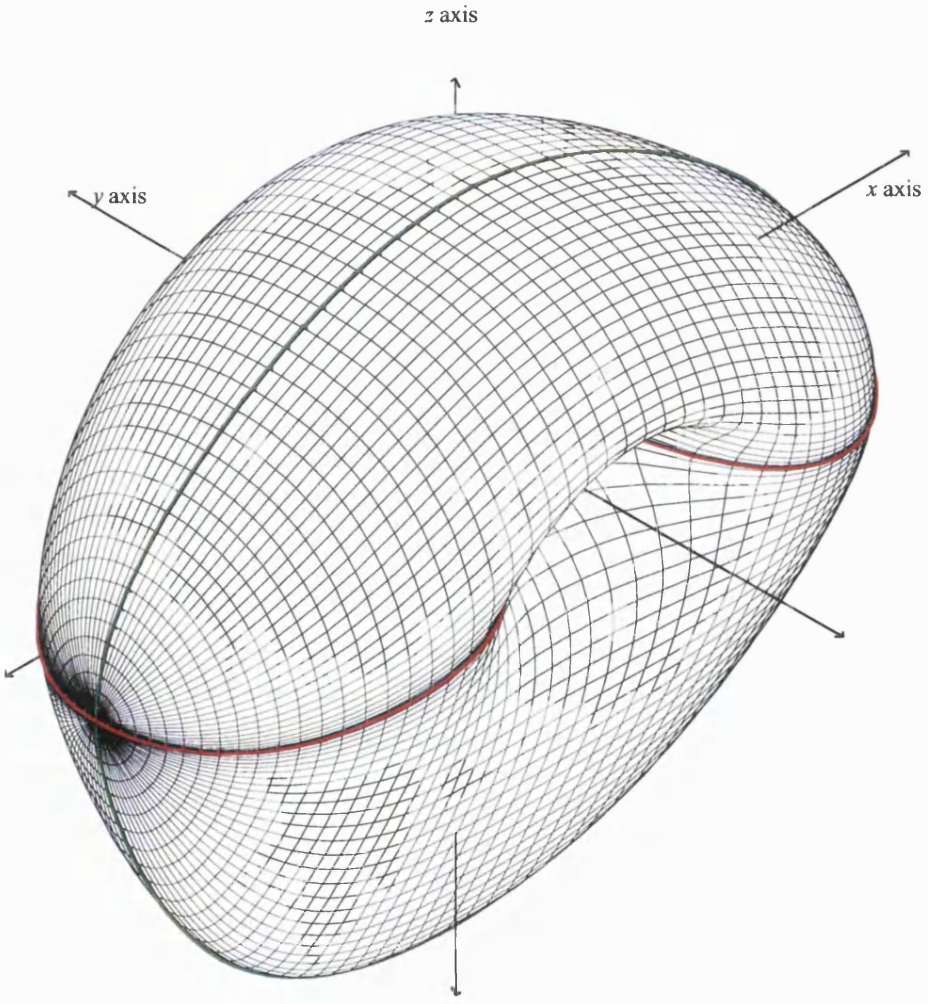


Figure 4.2.11 Axonometric view of deformed torus showing limits of ϕ

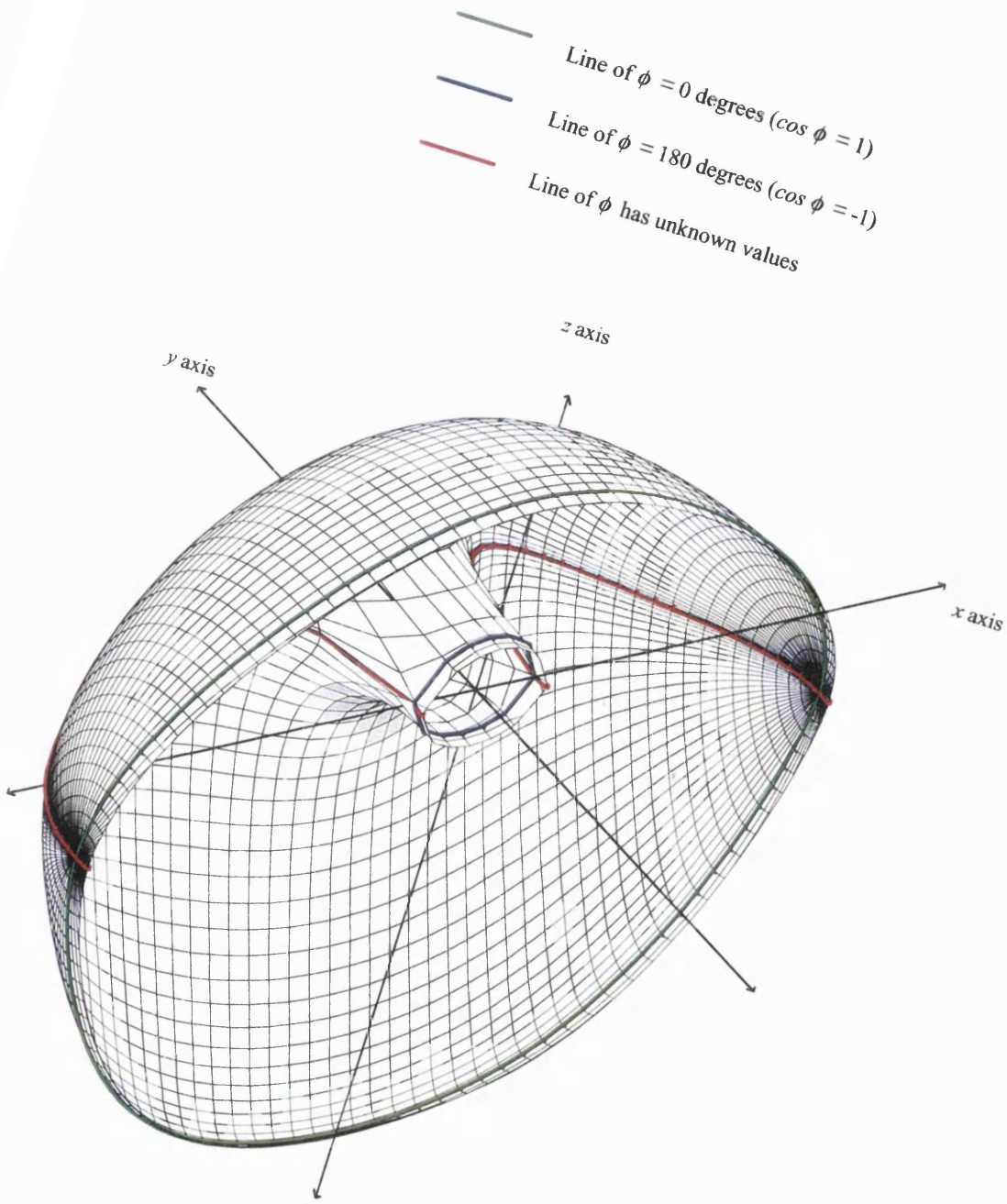


Figure 4.2.1m: Cut-away axonometric view of deformed torus showing limits of ϕ

The graph given in figure 4.2.1n describes the relationship between x and $\cos\phi$, and the equations given in (4), (5), (6) and (7) control the value of $\cos\phi$ at its boundary conditions. The variable, q, is used to represent $\cos\phi$ and the variable, Q, depends only upon the value of j and lies in the range 0 to 1.

At $Q = 0$, $\phi = 0$ degrees and

$$q = 1. \qquad (4)$$

Around the hole at $Q = 1$, $\phi = 180$ degrees, therefore $q = -1$, so that,

$$q + 1 = 0. \qquad (5)$$

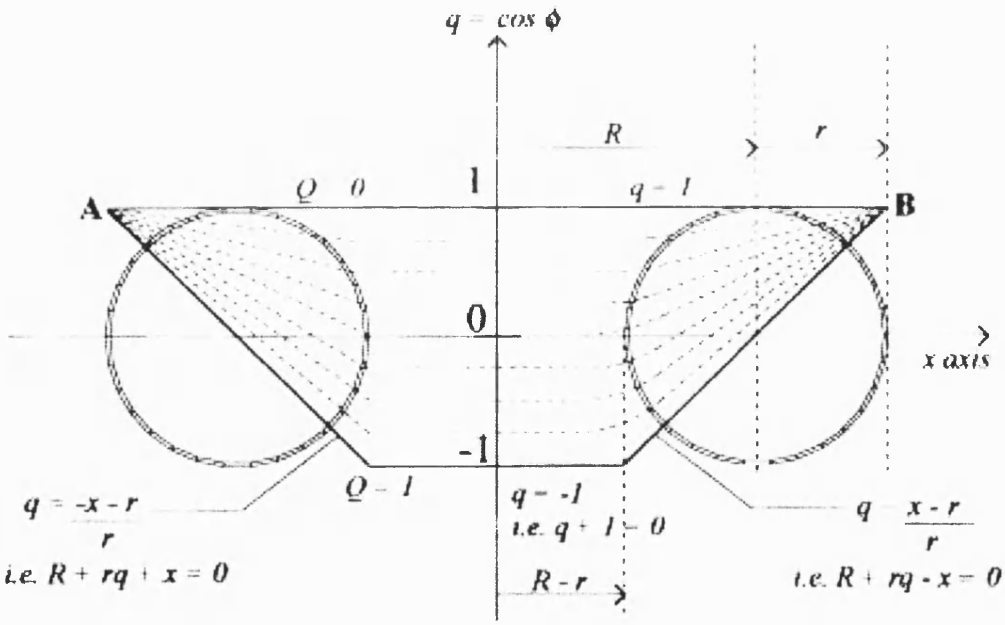


Figure 4.2.1n – Graph showing relationship of q and x

Beyond the hole, in the zone where the horizontal plane cuts the torus surface the maximum value of ϕ, corresponding to the minimum value of $\cos\phi$, is given by, $q = +\left(\frac{x-R}{r}\right)$, when x is positive, so that,

$$R + rq - x = 0, \qquad (6)$$

and, $q = -\left(\frac{x-R}{r}\right)$, when x is negative, so that,

$$R + rq + x = 0. \qquad (7)$$

Thus when $Q = 0$ the minimum value of q is given by (5), (6) or (7) as appropriate in which the value of x is given by (3).

When Q is greater than 0, but less than 1, combining the expressions in (5), (6) and (7) gives the function, f, in (8), which is used to determine the values of $\cos\phi$ in the intermediate regions of the torus.

$$f = (q+1)\left[(R+rq)^2 - x^2\right] - \lambda = 0\ , \qquad (8)$$

where λ is given by the following relationship,

$$\lambda = 2(1-Q)\left[(R+r^2) - x^2\right]. \qquad (9)$$

Incorporating (9) into (8), we obtain,

$$f = (q+1)\left[(R+rq)^2 - x^2\right] - 2(1-Q)\left[(R+r^2) - x^2\right] = 0\ . \qquad (10)$$

It is necessary to introduce the expression for λ given in (9), in order to provide for all the intermediate values of Q for which values of $\cos\phi$, are sought. The expression for λ given in (9) is also to force the lines of q to fan through points A and B shown in figure 4.2.1n, because at A and B, $x = \pm(R+r)$ and $q = 1$, therefore $f = 0$ for all values of Q. The lines are forced to fan through points A and B, simply to create a pattern of lines that merge at the torus edge, giving an attractive result.

Solving *f* using *f*dashed

In attempting to solve the expression in (10) for q, the function develops into a cubic, which at the time of the writing of the program was erroneously believed not to have a standard solution using analytical methods. The footnote in section 4.1.1.2 gives the

general analytic solution of a cubic. Therefore Newton's method, using the repeated algorithm

$$q_{n+1} = q_n - \frac{f(q_n)}{f'(q_n)} \qquad (11)$$

in which f' is the first derivative of f is adopted, from which solutions for all values of q are obtained, which produce the smoothly curved lines on the torus surface shown in figure 4.2.1e, h, i, k and l. Newton's method is used again later in several of the other design studies, and the reader is referred to the study for the footbridge using branching methods given in section 4.1.1.2, where the method is explained in more detail.

Having set values of x according to the required intervals between the ribs, and having solved (10) for values of q, figure 4.2.1p shows the sequence according to which all remaining co-ordinates at a given slice are obtained.

The reader is referred to the computer program in Appendix C1.3, which gives full details of the code to generate the detailed torus. References to Staad that occur throughout the computer program relate to data generation for structural analysis purposes using **Staad**, the Structural Analysis and Design package, and are not dealt with in this dissertation.

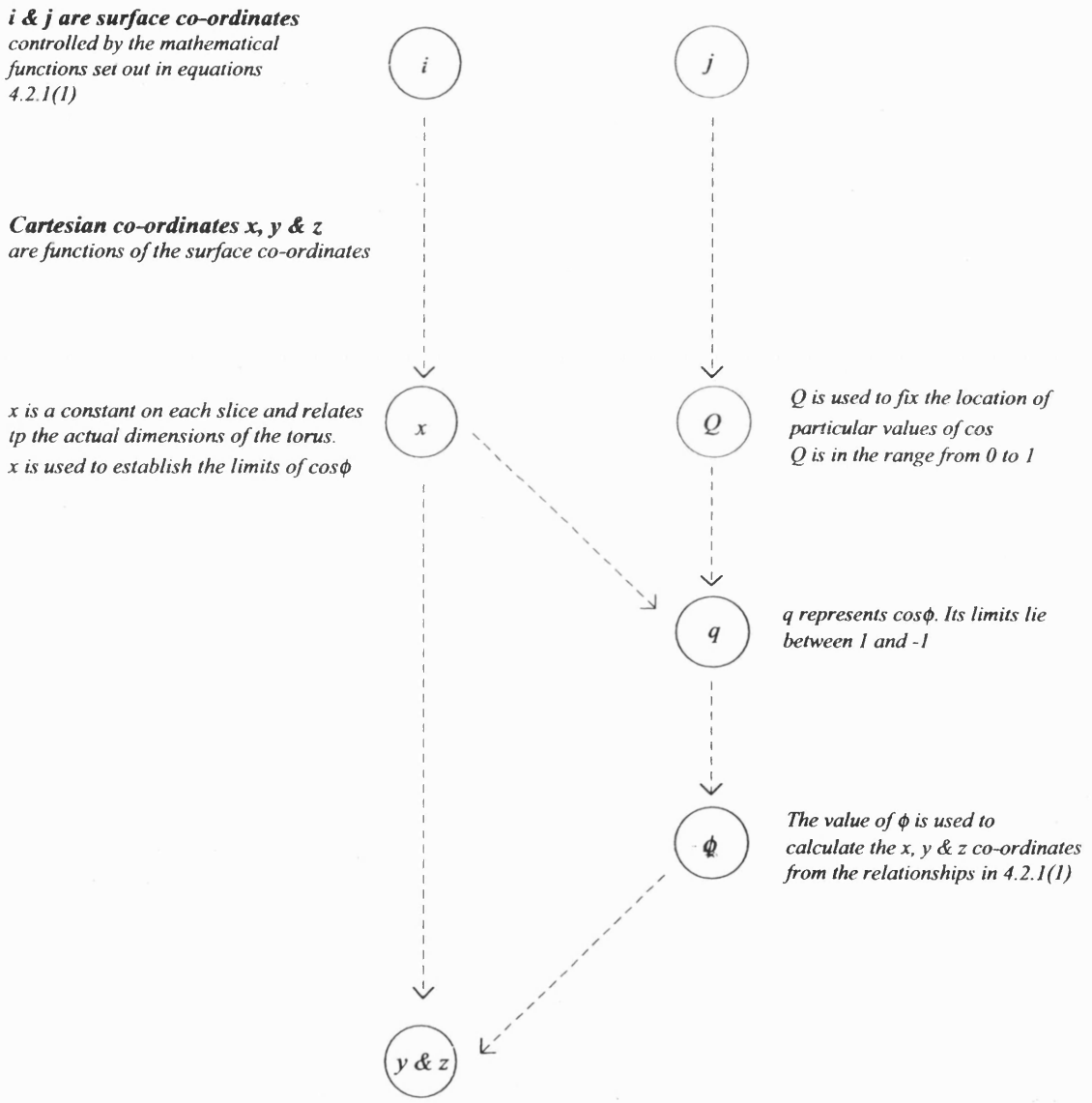


Figure 4.2.1p Sequence diagram showing how values of *x*, *y*, ϕ and *z* are determined

Deforming the Torus

All steps to deform the torus arose in direct response to modifications required by the architects, some examples of which are shown in the sketches in figure 4.2.1q.

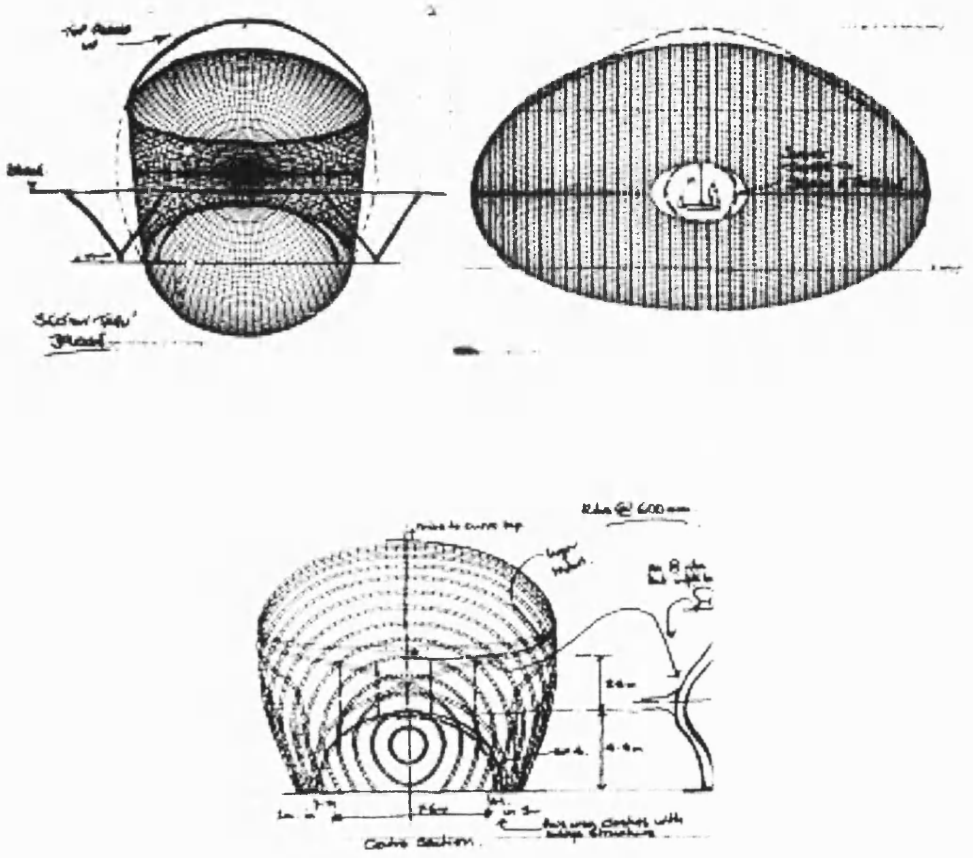


Figure 4.2.1q Sketches from RRP showing required modifications to torus geometry

Co-ordinates of x are determined according to the method described in the section entitled, 'Determining the rib intervals' and the functions to deform the torus have no effect on values of x. The torus is deformed in eleven successive steps, which distort the y and z co-ordinates only as can clearly be seen from the section diagrams in figures 4.2.1r and 4.2.1s. Figure 4.2.1t shows the corresponding axonometric diagram relating the deforming torus.

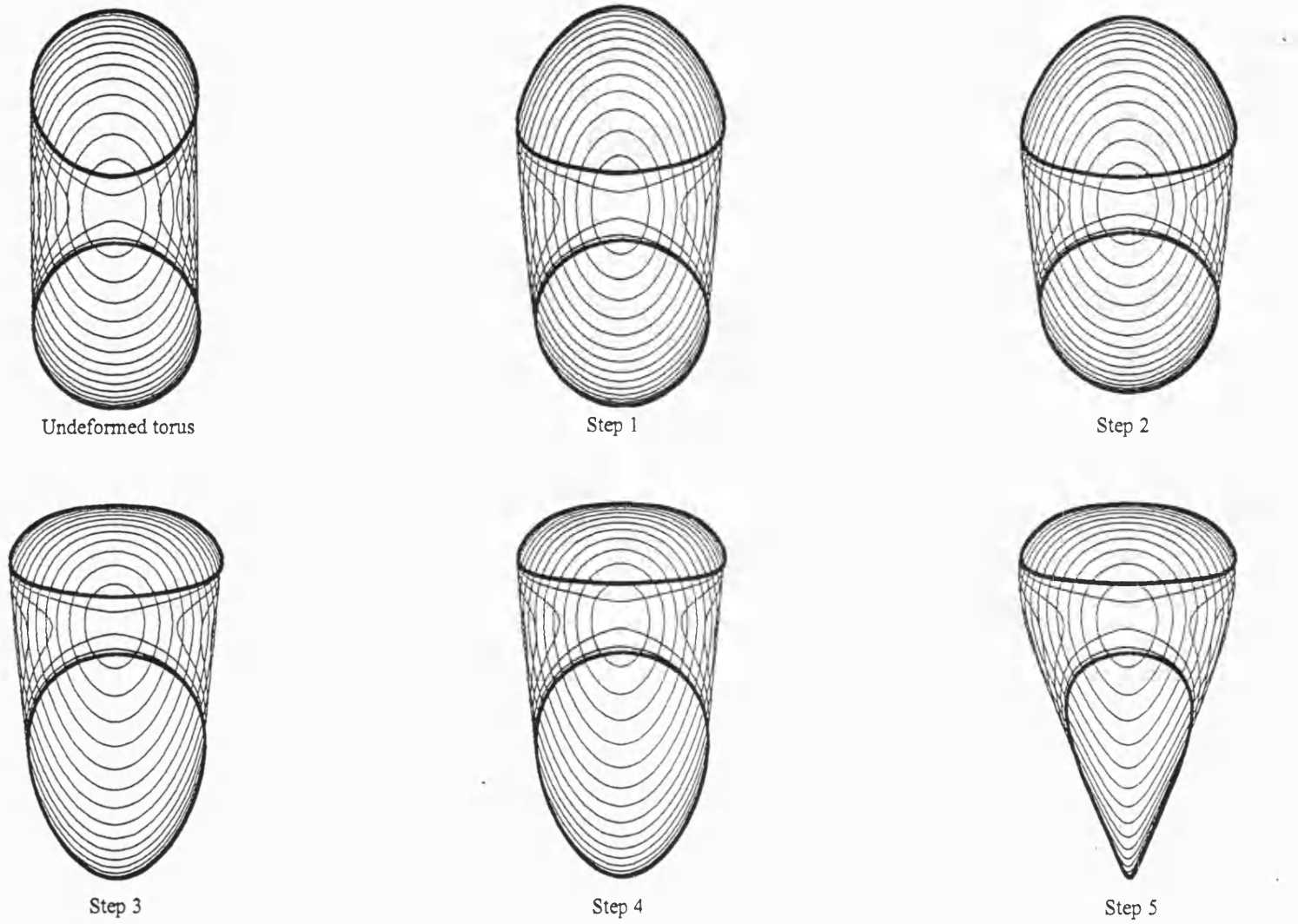


Figure 4.2.1r Deforming the torus, steps 1-5

Step 6

Step 7

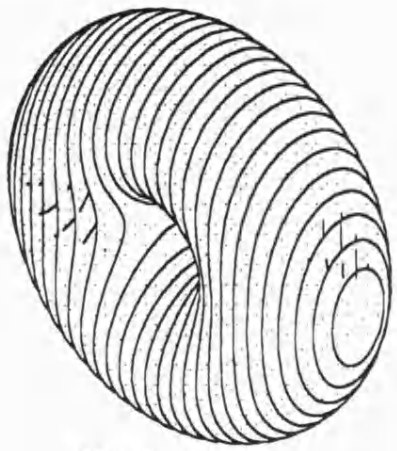
Step 8

Step 9

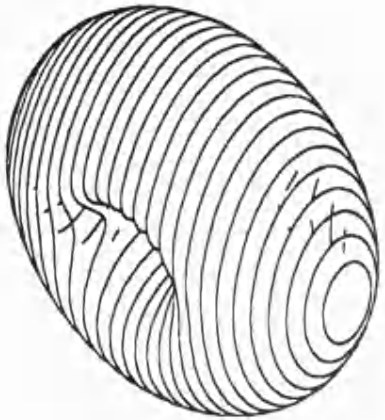
Step 10

Step 11

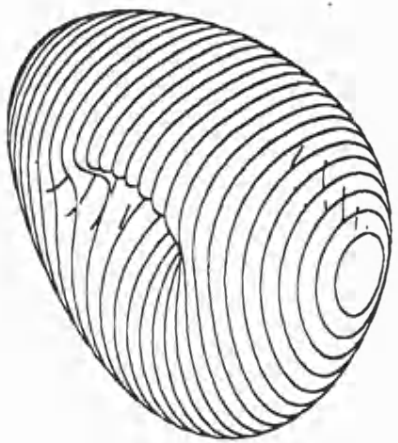
Figure 4.2.1s Deforming the torus, steps 6 - 11



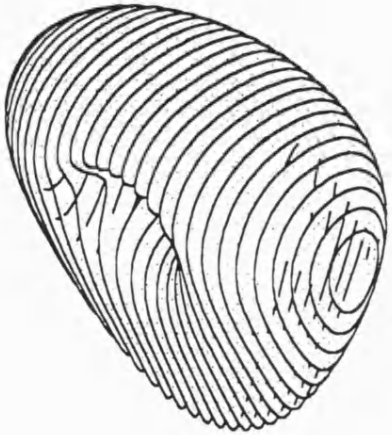
Undeformed torus



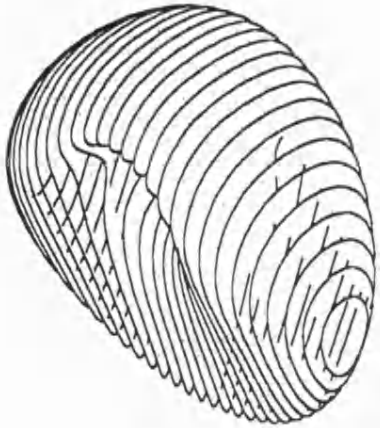
Step 2



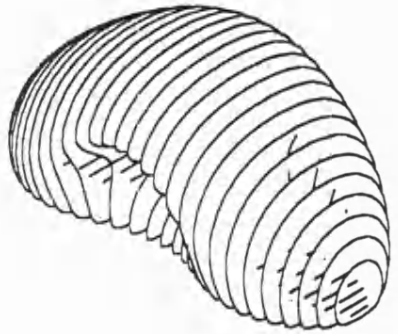
Step 4



Step 5



Step 9



Step 11

Figure 4.2.1t - Axonometric views of the deforming torus

In step 1to deform the torus, y is set equal to

$$y\left(\frac{1-q}{2}\right)^3\left(1.2+\sin\theta\right)+0.6\left(R+r\right)\sqrt{1-\left(\frac{x^2+z^2}{\left(R+r\right)^2}\right)}\left[1-\left(\frac{1-q}{2}\right)^3\left(1.2+\sin\theta\right)\right],$$

which stretches the y co-ordinates nearest the centre of the torus, to give it a more bulbous appearance. In general, the terms q, x, and z are included in this function to make the distortion of y dependent on the values of all other co-ordinates. As q, x, and z diminish or increase in their values, there is a corresponding effect on y. Writing the functions in this way maintains the overall coherence of the form, and relates the distortions to the geometry of the torus.

In step 2, z is set equal to $z-0.4\left(R-r\right)\tanh\left(0.5\frac{z}{\left(R-r\right)}\right)$,

which has the effect of flattening the torus around the centre. Where values of z are close to zero through the centre of the torus, the right-hand side of this function has a near zero value, which means that the effect of this shrinking diminishes accordingly. In the diagram of step 2 in figure 4.2.1r, the overall height dimensions of the torus have been evenly compressed.

In step 3 of the deformation process, z is set equal to $z-0.4\frac{z^2}{R+r}$,

which shrinks positive values of z significantly due to the squaring of z. When z is negative, however, it has the effect of increasing the magnitude of z, which means that the lower half of the torus is very much elongated as shown in step 3 of figure 4.2.1r.

In step 4, $0.03\left(R+r\right)\left(\frac{1-q}{2}\right)^6\left(1+\sin\theta\right)$ is added to z.

This magnifies the compression of z at the centre of the torus by a very small amount, where the values of $\left(\frac{1-q}{2}\right)^6$ are biggest.

In step 5, $0.3\frac{yz}{\left(R+r\right)}\left(1-\sin\theta\right)$ is added to y.

The effect of this is greatest in the lower sector of the torus, where values of y are initially small. Adding to y in the lower sector of the torus equates to *subtracting* from it, because y is being multiplied by a negative value of z in the lower sector. This results in the initial value of y diminishing significantly, which increases the taper of the bottom sector of the torus as seen in step 5 of figure 4.2.1r.

In step 6, $0.2y\left(\frac{1-\sin\theta}{2}\right)^5$ is subtracted from y,

and the effect of the tapering in step 5 is further accentuated, but only to a small degree.

In step 7, $0.3\frac{x^2}{R+r}$ is subtracted from y,

which results in y diminishing towards the outer edges of the torus, as x increases.

In step 8, 6100mm is added to z and in step 9, y is set to $0.97y$ in order to shrink the overall width of the torus.

In step 10, $0.2\frac{x^2}{R+r}$ is subtracted from z, so that z decreases with x.

In step 11, $0.2\frac{x^4}{(R+r)^3}$ is added to z. In addition z was set equal to $\frac{z+\sqrt{z^2+c^2}}{2}$

where c is a constant length. This has the effect of raising all the torus below ground level.

Sector Partitioning

Before concluding the description of the geometry-finding process of the Rest Zone, an important aspect of the programming strategy is outlined and illustrated below, relating to the partitioning of the torus into sectors. The reason for the partitioning is to create a suitable strategy for managing the order of calculating the co-ordinates.

By introducing terms such as *toporbottom*, *backorfront*, Parts numbered 0 and 1 and Sectors, numbered from 0 to 3, all of which are declared at the start of the program, the functions involved in the program computation only operate on the relevant Part and Sector of the torus, which is specified by the condition attached to that function.

The diagram in figure 4.2.1u explains the logic of the quadrant geometry. The order in which the calculations are performed enables the correct shape and alignment of lines on the torus to be drawn.

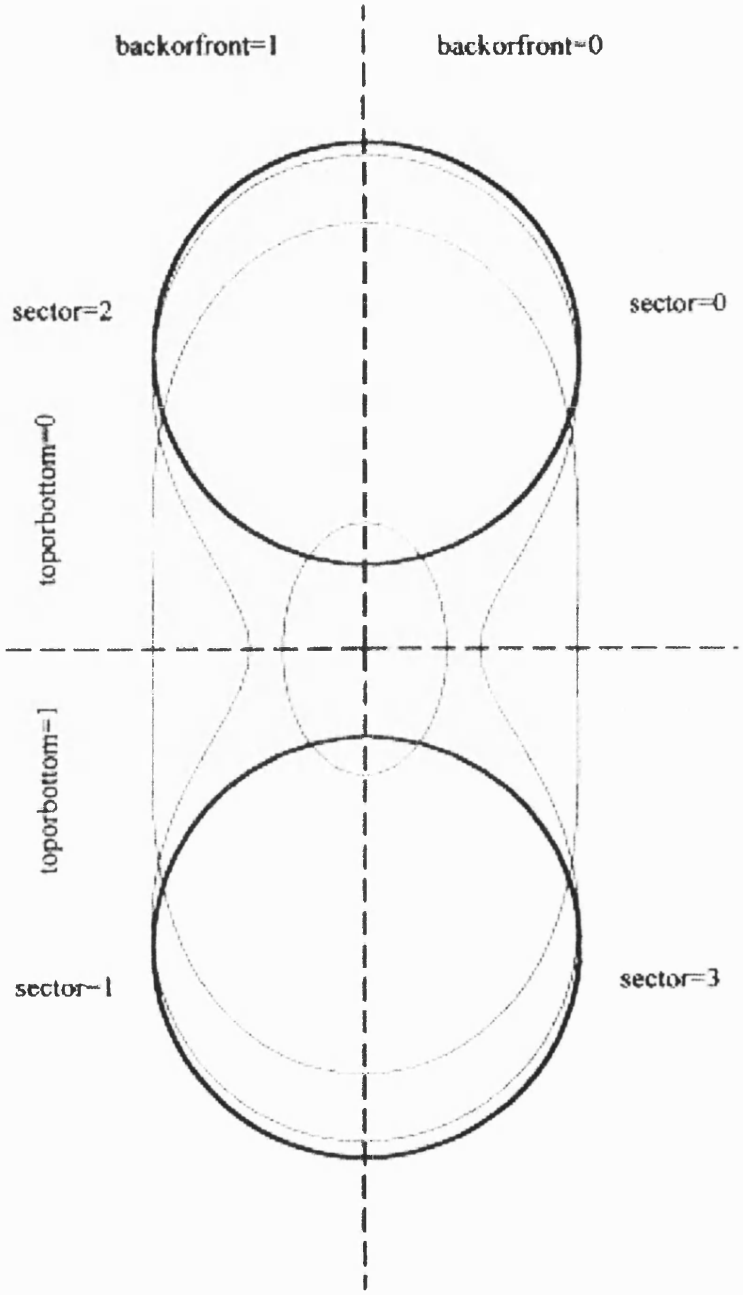


Figure 4.2.1u – Sector partitioning of torus

In the computer program, Part = 0 refers to the outer cladding of the torus and Part = 1 refers to the inner cladding of the torus. For Sector = 0 or 2 and toporbottom = 0, the co-ordinates of the top half of the torus are computed, whilst for Sector = 1 or 3 and toporbottom = 1, the co-ordinates of the bottom half of the torus are computed.

Chapter 5.0 Summary and Conclusions

5.1 Summary

Chapter one discusses the philosophy and history of organicism and the mechanisms by which it came to affect design in architecture. It explores the development of ancient enquiry and its subsequent evolution into the fields of natural science and morphological study of the seventeenth and eighteenth centuries. It concludes with a discussion of the impact of organicism on the decorative art and architecture of the late nineteenth century.

The rich impetus of natural historians and morphologists provided architects with a design vocabulary with which to shape one particular tradition, the *Art Nouveau.*

An organic tradition developed separately under Louis Sullivan and Wright, amidst others in North America, which occurred as a result of the affiliation of Sullivan and his predecessors with the Beaux Arts tradition in Paris.

Chapter two presents the graphic and geometrical characteristics of form in nature, both at the microscopic and macroscopic scale, in order to suggest a possible template for design. It emphasises the features of structure, curvature, spiralling and movement.

Chapter three discusses mathematical methods for constructing curved geometry and develops the numerical and geometrical tools to assist with the generation of form in architecture.

Chapter four draws on the theory and methodology provided in earlier chapters to demonstrate the synthesis between organicism as a way of thinking and about mathematics as a way of doing. This synthesis is achieved in the realisation of the design studies.

The techniques investigated in this chapter explore opportunities to harness the potential of current technological capacity in the generation of complex shape information. The integration of mathematical methods into the process of form description offers a cohesive approach to design, in which the provision of accurate data for analysis and automatic machining is made possible.

Above all, this work has set out to record an investigation into alternative ways of generating form with the aim of making its principles available to a wider public. It aims to expand the scope of conventional design tools, and to throw light on what may initially be

considered to be a clinical approach to architectural design, which has had unexpected and worthwhile results. Underlying the technique is the desire to apply a new expression of organicism to architecture using today's technology.

5.2 Conclusions

An inter-disciplinary work of this nature has advantages and disadvantages. Several problems arise in attempting to apply principles that originate in other disciplines to the generation of form in architecture using natural form as the frame of reference.

The first is one of reluctance on the part of an architectural readership to become acquainted with methodologies that lie outside the conventional boundaries of their discipline, which at first seem impenetrable. Similarly, the cross-pollination of one subject methodology with another carries with it the danger of over-simplification and abstraction from its original source and application.

The second problem is that the decision to adopt very precise methods of describing geometry bears with it the scope for incompatibilities to exist between extremely accurate data on the one hand and the limitations of manual production methods on the other. Automatic machining is well advanced in the industrial sector where high initial investment is counterbalanced by the economies of scale that result from mass-production. Clearly, mass production is not always an appropriate or desirable option in architecture.

The techniques are limited in the sense that they are not suited to all types of design. Buildings which are made up of heterogeneous parts do not lend themselves well to the methods, since the purpose of a single function to control the shape of a building is defeated when the function loses its elegance by becoming convoluted.

The methods are however suited to the description of coherent objects such as bridges, or to the geometrical definition of single aspects of a particular design as is well demonstrated in the example in Appendix D of the British Museum grid shell roof.

Despite the problems, it is hoped that by extending the set of tools available to the designer, new possibilities are created which lie beyond those offered by conventional drawing methods and techniques.

The study is primarily aimed at breaking the apparent barriers between mathematical theory and artistic expression with a view to presenting it to an audience of designers who may be interested to begin experiments using new approaches to the generation of form in architecture and engineering design.

Bibliography

Texts on the natural form and geometry

Alexander, Christopher, 'Notes on the synthesis of form', Harvard, 1964.

Bonner, John Tyler, 'Morphogenesis, an essay on development', Atheneum, 1952, 1963.

Bronowski in **Kepes, Gyorgy ed.,** 'Structure in Art and Science, Studio Vista, London 1965.

Coexeter, H.S.M., 'Geometry Re-visited', L.W.Singer & Co., New York, 1967.

Coexeter, H.S.M., 'Non-Euclidean Geometry', University of Toronto Press,.Toronto, 1957.

Coexeter, H.S.M., 'Introduction to Geometry', John Wiley & Sons Inc.USA, 1961.

Cook, Theodore Andrea., 'The Curves of Life', Spirals in Nature and Art based on the Manuscripts of Leonardo da Vinci, Constable & Co., London 1914 and Dover, New York, 1979.

Cook, Theodore Andrea, 'Spirals in Nature and Art: a study of spiral formations based on the Manuscripts of Leonardo da Vinci, John Murray, London 1903.

Cundy, H.M. and Rollet, A.P., 'Mathematical Models, Oxford University Press, London, 1952.

D'Arbeloff, Natalie, 'Designing with Natural Forms' B.T.Batsford, London 1973.

D'Arcy Wentworth Thompson, 'On Growth and Form', An abridged edition edited by Bonner, John Tyler. Cambridge University Press, 1961.

Day, L.F., 'Nature and ornament; nature the raw material of design, ornament its finished product, B.T.Batsford, London 1909.

Faux, I.D, and Pratt, M.J., 'Computational Geometry for Design and Manufacture', (Ellis Horwood, 1979).

Feininger, A., 'The Anatomy of Nature', Crown, New York, 1956.

Feininger, A., 'Forms of Nature and Life', Viking, New York, 1966.

Field, M. and Golubitsky, M., 'Symmetry in chaos, a search for pattern in mathematics, art and nature', Oxford University Press, 1992.

Ghyka, M., 'The Geometry of Art and Life', Sheed and Ward, New York, 1949.

Hale, Nathan Cabot., 'Abstraction in Art and Nature', Watson Guptill Publications, New York, 1972.

Häckel, Ernst Heinrich., 'Art Forms in Nature', Dover Publications Inc., New York, 1974.

Hertel, Heinrich., 'Structure, Form, Movement', Reinhold, New York, 1966, 1963.

Hildebrandt, S. and Tromba, A., 'Mathematics and Optimal Form', Scientific American Books Ltd., 1985.

Hogarth, W., 'The Analysis of Beauty', Samuel Bagster, 1822.

Holt, Michael., 'Mathematics in Art', Studio Vista, London, Van Nostrand Reinhold Co., N.York 1971.

Jöedicke, J., 'Shell Architecture', Reinhold, New York, 1963.

Kepes, Gyorgy ed., 'Module, Symmetry, Proportion' Studio Vista, London 1966.

Kepes, Gyorgy ed., 'Structure in Art and Science, Studio Vista, London 1965.

Latham, W. & Todd, S., 'Evolutionary Art and Computers', Academic Press Ltd. London, 1992.

Lord, E.A. & Wilson, C.B., 'The Mathematical description of shape and form', (Ellis Horwood, 1984).

Mach, E., 'Space and Geometry', Opencourt, La Salle, Illinois, 1960.

Mandelbrot, Benoit., 'The fractal geometry of nature', W.H.Freeman, San Fransico 1982.

McHarg, I. L., 'Design with Nature', John Wiley and Sons, New York 1992.

Padovan, R., 'Proportion: Science, Philosophy, Architecture', E & FN Spon, London 1999.

Patterson, E.M., 'Topology', Oliver and Boyd, London and Edinburgh, 1956.

Pearce, Peter., 'Structure in nature is a strategy for design', MIT Press, Cambridge, MA, 1978.

Pescarmona, L.M., 'Ceramic Forms derived from Natural Forms', MFA Thesis, George Washington University, Washington, 1977.

Pettigrew, James Bell., 'Design in Nature' Volumes I, II and III, Longmans Green & Co. Lon, 1908.

Reid, G.W. (ASLA), 'From Concept to form in Landscape Design', Van Nostrand Reinhold, N.York, 1993.

Robinson, R.G., 'Natural Forms as a basis for Sculpture' (Master's report), San Diego Sate University, 1964.

Siegel, C., *Strukturformen* translated by Burton, T.E, 'Structure and Form in Modern Architecture' trans. Crosby, Lockwood & Son Ltd., 1962.

Simmons, G.F., 'Introduction to Topology and Modern Analysis', McGraw-Hill Book company, inc., 1963.

Skoog, West, Hollen and Crouch, 'Analytical Chemistry – an introduction', Harcourt College publishers, Harcourt Inc., 2000.

Sommerville, D.M.Y., 'The Elements of non-Euclidean Geometry, Dover, New York, 1958.

Stevens, P.S., 'Patterns in Nature', Little, Brown & Co., USA, 1974, Peregrine Books 1976.

Stewart. Ian, 'Does God Play Dice?: the mathematics of chaos', Basil Blackwell Ltd., UK, 1989.

Stix, H., Stix, M. and Abbott, R.T., 'Shell: Five hundred million years of inspired design', Abrams, New York, 1969.

Strache, Wolfe., 'Forms and Patterns in Nature', Pantheon, 1973.

Struik, D.J., 'Lectures on Classical Differential Geometry', Addison-Wesley Publishing Co., 1961.

von Seggern, D., 'CRC Standard curves and surfaces', CRC Press, 1993.

Tames, R., 'William Morris - An illustrated life of William Morris 1834 - 1896', Shire Publications, London 1972.

Watkinson, R., 'William Morris as Designer', Studio Vista, London 1967/79.

Warner, J., 'Studies in Organic Morphology', J.B.Lipincott & Co., 1857.

Wells, D., 'The Penguin Dictionary of curious and interesting Geometry, Penguin, 1991.

Whittaker & Watson, 'Modern Analysis', Cambridge University Press, 1936.

Weller, H.R., 'Nature and design: Illustrative steps and examples of pattern and ornament and design based on natural forms' Charles and Dible, London 1909.

Texts on Art Theory, Natural Philosophy and the Philosophy of Science

Adorno, T., 'Aesthetic Theory', The Althone Press, London, 1997.

Wieland in **Barnes, J., Schofield, M., Sorabji, R.,** 'Articles on Aristotle, Gerald Duckworth and Co., 1975

Barnes, J., Schofield, M., Sorabji, R., 'Articles on Aristotle, Gerald Duckworth and Co., 1975

Blackburn, S., 'The Oxford Dictionary of Philosophy', OUP, 1996.

Cajori, F, Mathematical Principles of Natural Philosophy and the System of the World, University of California Press (Berkeley: 1934).

Clavelin, M., 'The Natural Philosophy of Galileo', MIT Press, 1968, *trans.* 1974.

Drake, S., 'Galileo', Oxford Univesity Press, 1980.

Drake, S., (trans.) 'Two New Sciences including centres of gravity and force of percussion', Galileo (1968), University of Wisconsin Press, *trans.* 1974.

Heath, T., 'Mathematics in Aristotle', Clarendon Press, Oxford, 1949, 1970.

Koestler, Arthur. 'The Act of Creation' Pan Books Ltd., London 1970.

Langer, Susanne, 'Feeling and Form - A theory of Art developed from philosophy in a new key, Routledge and Keegan Paul, New York, 1953.

Losee, J., 'A Historical Introduction to the Philosophy of Science', Oxford University Press, 1993.

MacCurdy, E., 'The Notebooks of Leonardo da Vinci', the reprint Society, 1954.

Serres, M., (ed.), *translated from the French,* 'A History of Scientific thought', Blackwell, Oxford, 1995.

Sheppard, A., 'Aesthetics: an introduction to the philosophy of art, Oxford University Press, 1987.

Whyte, L. L. (ed)., 'Aspects of Form', Lund Humphries, London 1968.

Whyte, L. L., 'Accent on Form', Harper, New York, London, 1954.

Texts on Architecture and Nature

Arts Council of Great Britain, 'Le Corbusier Architect of the Century', 1987.

Baker, G.H., 'Le Corbusier, an Analysis of Form, Van Nostrand Reinhold, 1989.

Blake, P., 'Le Corbusier – Architecture and Form', Penguin Books, 1960.

Bolon, C., Nelson, R.S., Seidel, L., 'The Nature of Frank Lloyd Wright', The University of Chicago Press, Chicago, 1988.

Brooks, H.A. (ed.), 'Le Corbusier 1887 – 1965', Electa, 1987.

Brooks, H.A., 'Le Corbusier's formative years', University of Chicago Press, 1997.

Dernie, D., Carew-Cox, A., 'Victor Horta', Academy Editions, London 1995.

Drew, Philip., Frei Otto - Form and Structure', Lockwood Stopes, London 1976.

Dunster, D., 'Hector Guimard', Architectural Monographs and Academy Editions, London, 1972.

Elia, M.M., 'Louis Henry Sullivan', Electa, 1995, & Princeton Architectural Press, 1996.

Faber, C. 'Candela - The Shell Builder', The Architectural Press, London, 1963.

Fondation Le Corbusier and The Architectural History Foundation, 'Le Corbusier Sketchbooks 1 (1914 – 1948)', AHF, New York, MIT Press, Cambridge, Massachusetts and FC, Paris 1981.

Frazer, J., 'An Evolutionary Architecture', The Architectural Assoc., London 1995.

Hoffmann, D., 'Frank Lloyd Wright, architecture and nature', Dover 1986.

Huxtable, A.L., 'Pier Luigi Nervi', Mayflower, London, 1960

Institute for Lightweight Structures (IL). 'IL3 Biology and Building Part I'.

Institute for Lightweight Structures (IL). 'IL4 Biology and Building Part II'.

Institute for Lightweight Structures (IL). 'IL6 Biology and Building Part III'.

Jencks, C., 'The architecture of the jumping universe; a polemic: how complexity science is changing architecture and culture', Academy Press, London 1995.

Jencks, C., 'Frank O. Gehry: individual imagination and cultural conservatism',

Academy Press, London 1995.

Le Corbusier, 'Ronchamp', Les Editions Girsberger, Zurich, Verlag Gerd Hatje, Stuttgart, 1957.

Levine, N., 'The Architecture of Frank Lloyd Wright', Princeton University Press, 1996.

Marks, R., and Fuller, R.B., 'The Dymaxion world of Buckminster Fuller', Anchor Books, Garden City and New York, 1973.

McCarter, R., (ed.), 'Frank Lloyd Wright: A primer on architectural principles', Princeton Architectural Press, New York, 1991.

Moos, Stanislaus von, 'Le Corbusier, Elements of a Synthesis', MIT 1979.

Nervi, P.L., (translated by Giuseppe Nicoletti), 'New Structures', The Architectural Press, London, 1963.

Nicholson, Patricia, Coyle, (ed.) with a biography by Iovanna Lloyd Wright, 'Architecture: man in possession of his earth', Macdonald, London, 1963.

Otto, F. and Rasch, B., 'Finding Form: towards an architecture of the minimal', Axel Menges, Stuttgart, 1996.

Pawley, M., 'Frank Lloyd Wright I Public Buildings', Thames and Hudson, London, 1970

Pawley, M., 'Buckminster Fuller', Trefoil Publications Ltd, London, 1990

Pauly, D., 'Le Corbusier: La Chapelle de Ronchamp', FLC, Birkhauser, 1997

(translation from French into English: Sarah Parsons, Paris).

Pevsner, N., 'Pioneers of modern design from William Morris to Walter Gropius',

Penguin Books, 1975

SFB 230. 'Evolution of Natural Structures', University of Stuttgart, 1994.

Sullivan, Loius. H., 'A system of Architectural Ornament according with a philosophy of man's powers, AIA Press, 1924.

Sullivan, Loius. H., 'Kindergarten Chats', New York, 1947

Steadman, P., 'The Evolution of Designs - Biological analogy in architecture and the applied arts', Cambridge University Press, 1979

Steadman, P., 'Architectural Morphology', an introduction to the geometry of building plans, Pion Ltd., London, 1983.

Torroja, E., 'The structures of Eduardo Torroja', T.W.Dodge Corporation, New York, 1958.

van Eck, C., Organicism in nineteenth-century architecture: *An enquiry into its theoretical and philosophical background.* Architecture & Natura Press, Amsterdam 1994.

Wittkower, R., 'Architectural principles in the age of humanism, *2nd ed.* London,1952

Zevi, Bruno, 'Towards an organic architecture', Faber & Faber, London, 1950.

Journal Articles

Adams, D., 'Rudolf Steiner's first Goetheanum as an illustration of organic functionalism' in Journal of the Society of Architectural Historians, vol. 51, June 1992, pp 182 - 204.

Architectural Review (11 article special section), 'German organic approach to architecture', vol. 177, June 1985, pp 25 - 89.

Bornstein, E., 'Structure and colour in nature: toward symbiosis in art and architecture (can artists and architects still learn from nature?), The Structurist no. 31/32, 1991-1992, p 96 - 105.

Caldwell, A., 'Nature and Architecture' in The Structurist no. 23/24, 1983-1984,p 34 - 39.

Candela & Calatrava, 'Two generations of engineering architecture' in World Architecture, Issue 13, 1992?

Eck, Caroline van, 'Goethe and Alberti: organic unity in nature and architecture', in The Structurist no. 35-36, 1995-1996,p 20 - 26.

Eck, Caroline van.; Padovan, R., (reviewer), 'Organicism in nineteenth century architecture'(book review), in The Architectural Review vol. 196, July 1994, p 96 - 97.

Eck, Caroline van.; Thistlewood, D., (reviewer), 'Organicism in nineteenth century architecture', in The British Journal of Aesthetics, vol. 35, Oct. 1995, p 407 - 409.

Gorlin, A., 'Geometry and nature in the work of Frank Lloyd Wright' in A + U, no. 185, February 1986, pp 55 - 62.

Hoffmann, D.; Andi, S., (reviewer), 'Frank Lloyd Wright, architecture and nature (book review), in Domus no.682, April 1987, p iv.

Leatherbarrow, D., 'Architecture and situation: a study of the architectural writings of Robert Morris' in Journal of the Society of Architectural Historians, vol. 44, March 1985, pp 48 - 59.

Mumford, M., 'Form follows nature: the origins of American organic architecture' in Journal of Architectural Education, vol. 42, Spring 1989, pp 26 - 37.

Pauly, D., 'Il segreto della forma. (Le Corbusier and natural forms)' in Casabella vol 51, Jan/Feb. 1987, pp 86 - 93.

Saddy, P., 'La ricchezze della natura. (Le Corbusier's thoughts on architecture, landscape and nature)' in Casabella vol 51, Jan/Feb. 1987, pp 42 - 51.

The Structurist no. 35-36, 1995-1996,p 4 - 115, 'From the mechanical to the organic: the constructive principle in art and architecture (18 article special issue).

Weingarden, L. S., 'The colours of nature Louis Sullivan's architectural polychromy and nineteenth-century colour theory' in Winterthur Portfolio, vol. 20, 1985, pp 243 - 260.

Weingarden, L. S., 'Naturalized nationalism: a Ruskinian discourse on the search for an American style of architecture' in Winterthur Portfolio, vol. 24, 1989, pp 43 - 68

Wissinger, J., 'Organicism: a neglected tradition (storefront for Art and Architecture, New York)' in Progressive Architecture vol. 66, March 1985, p. 28.

Texts on Movement and Architecture

Blaeser, W., 'Santiago Caltrava'

Breshears, J.E., 'Tools and Technology Body and World: a structurally dynamic pedestrian bridge', Rice University, Houston, Texas, 1993.

Breshears, J.E., 'Tools and Technology Body and World: an exploration of technology transfer', Rice University, Houston, Texas, 1993.

Britvec, S.J., 'Stability and Optimization of Flexible Space Structures', Berkhäuser Verlag, Basel, Boston Berlin, 1995.

Bulson, P.S. ed, 'Rapidly Assembled Structures, Computational Mechanics Publications, Southhampton, UK and Boston, 1991.

Caltrava, S.V., 'Concerning the Foldability of Frameworks' Doctoral Thesis *(German Text)*, ETH, Zurich, Switzerland, 1981.

Caltrava, S.V., *'Il follo volo'*

Candela, F., 'Emilio Perez Piñero' in *En defensa del formalismo y otros escritos, Xtrait Ediciones,* 1985

Candela, F., 'Arquitectura Transformable, *Escuela Tecnica Superior de Arquitectura de Sevilla*, 1993.

Escrig, F. and Brebbia, C.A. ed., 'Mobile and Rapidly Assembled Structures II', Second International Conference on Mobile and Rapidly Assembled Structures MARAS '96, Computational Mechanics Publications, 1996.

Faegre, T., 'Architecture of the nomads', Anchor Books ed., Garden City, New York, Anchor Press/Doubleday, London 1979.

Faber, R.L., 'Differential Geometry and Relativity Theory', Marcel Dekker, Inc. New York, 1983.

Holton in **Kepes, Gyorgy ed.,** 'The Nature and Art of Motion', Studio Vista, London 1965.

Kepes, Gyorgy ed., 'The Nature and Art of Motion', Studio Vista, London 1965.

Kronenburg, R., 'Houses in Motion' The Genesis of the Portable Building, Academy Editions, 1995.

Kronenburg, R., 'Portable Architecture', Architectural Press, Oxford, 1996.

Muybridge, Eadweard. Lewis S. Brown ed., 'Animals in Motion, Dover Publications, inc. New York 1957.

Muybridge, Eadweard. Lewis S. Brown ed., 'The Human Figure in Motion', Dover Publications, inc. New York 1955.

Peitgen, H., Jürgens, H., Saupe, D., 'Chaos and Fractals, New Frontiers of Science', Springer-Verlag Inc., New York, 1992.

Pettigrew, James Bell, 'Animal Locomotion or Walking, Swimming, and Flying with a dissertation on Aeronautics. Henry S.King & Co. London, 1873.

Puertas, L., 'Demountable and Deployable Spatial Structures', Doctoral Thesis, *Bibliotecaria Escuela Tecnica Superior de Arquitectura, Universidad Politecnica de Madrid,* 1990.

Roland, F., 'Frei Otto: Structures', Longman, London, 1970.

Space Structures 4, Thomas Telford, London 1993.

Sweeney, J.J and Sert, J.L., 'Antoni Gaudi', The Architectural Press, London, 1970.

Tzonis, Alexander and Lefaivre, Liane, 'Movement, Structure and the work of Santiago Calatrava', Birkhauser, Basel, Berlin, Boston 1995.

Wyman, M. with **Sir C Bells** work: 'Animal Mechanics', 1827 & reprinted in 1902, Boston Society of Natural History, Cambridge Massachusettes, USA.

Zuk, William and Clark, Roger H., 'Kinetic Architecture' Van Nostrand Reinhold, New York 1970.

Texts on Structural Engineering

Topping, B.H.V & Sack, L. (ed.) Barnes, M.R. and Topping, B.H.V. (Special Issue Editors) 'Structural Engineering Review – Tension Structures Special Issue', Volume 6, Number 3-4, pages 143 – 256, August November 1994.

Appendix A1.0 Example computer programs

A1.1 Computer program to generate a Golden Section Spiral.

```
#include <fstream.h>
#include <iostream.h>

#include <math.h>

#define   MaxNo 5001

void DXFSetUp(void);
void DXFLines(void);
void DXFText(void);
void DXFFinishOff(void);

int   i,NoSegments,Layer,Colour,PointsPerCycle,NumberOfCycles;

float x (MaxNo],y[MaxNo],z[MaxNo],PI,
      r,r0,theta,phi,lambda,
      DXFx1,DXFy1,DXFz1,
      DXFx2,DXFy2,DXFz2;

ofstream Emma("Spiral1.dxf");

int main(void)
{
PI=4.0*atan(1.0);
DXFSetUp();
r0=0.5;

NumberOfCycles=6;

for(Layer=0;Layer<=1;Layer+=1)
{
if(Layer==0)PointsPerCycle=100;
else PointsPerCycle=4;

NoSegments=NumberOfCycles*PointsPerCycle;
```

```
lambda=(2.0/PI)*log((1.0+sqrt(5.0))/2.0);

phi=atan(exp(lambda*PI/2.0));
for(i=0;i<=NoSegments;i+=1)
{
theta=phi+(2.0*PI*i)/(1.0*PointsPerCycle);
r=r0*exp(lambda*theta);
x[i]=r*cos(theta);
y[i]=r*sin(theta);
z[i]=0.0;
}
```

```
Colour=Layer;
for(i=1;i<=NoSegments;i+=1)
{
DXFx1=x[i-1];
DXFy1=y[i-1];
DXFz1=z[i-1];
DXFx2=x[i];
```

```
DXFy2=y[i];
DXFz2=z[i];
DXFLines();
}
}
DXFFinishOff();
cout<<"DXF file written, end of program\n";
return 0;
}

void DXFSetUp(void)
{
Emma<<"0\n";
Emma<<"SECTION\n";
Emma<<"2\n";
Emma<<"ENTITIES\n";
}
void DXFLines(void)
{
Emma<<"0\nLINE\n8\n"<<Layer<<"\n";
Emma<<"10\n"<<DXFx1<<"\n";
Emma<<"20\n"<<DXFy1<<"\n";
Emma<<"30\n"<<DXFz1<<"\n";
Emma<<"11\n"<<DXFx2<<"\n";
Emma<<"21\n"<<DXFy2<<"\n";
Emma<<"31\n"<<DXFz2<<"\n";
Emma<<"62\n"<<Colour<<"\n";
}
void DXFFinishOff(void)
{
Emma<<"0\n";
Emma<<"ENDSEC\n";
Emma<<"0\n";
Emma<<"EOF\n";
Emma.close();
}
```

A1.2 Computer program to generate Multiple Inter-locking Spirals

This program generates multiple spirals firstly by rotating one spiral line in a clockwise direction from 0 and 360 degrees, then mirroring the spiral about the *y* axis and repeating the rotation in a counter-clockwise direction. The program which generated figure A1.2a is given below and a very similar program generated figure A1.2b. In each case the spiral pattern is based on a system of curvilinear rectangles.

```
#include <fstream.h>
#include <iostream.h>

#include <math.h>

#define   MaxNo 5001

void DXFSetUp(void);
void DXFLines(void);
void DXFText(void);
void DXFFinishOff(void);

int
i,NoSegments,Layer,Colour,Points
PerCycle,NumberOfCycles,
Spiral,NumSpirals,clockoranti,Ci
rcle,MyLine;

float
x[MaxNo],y[MaxNo],z[MaxNo],PI,
      r,r0,theta,lambda,
      DXFx1,DXFy1,DXFz1,
      DXFx2,DXFy2,DXFz2;

ofstream Emma("Spiral2.dxf");

int main(void)
{
PI=4.0*atan(1.0);
DXFSetUp();
```

```
NumberOfCycles=2;

NumSpirals=16;

PointsPerCycle=100;

Layer=0;
NoSegments=NumberOfCycles*Points
PerCycle;

lambda=(2.0/PI)*log((1.0+sqrt(5.
0))/2.0);

r0=40000.0/exp(lambda*2.0*PI*Num
berOfCycles);

for(clockoranti=-
1;clockoranti<=1;clockoranti+=2)
{
for(Spiral=1;Spiral<=NumSpirals;
Spiral+=1)
{
for(i=0;i<=NoSegments;i+=1)
{
theta=(2.0*PI*i)/(1.0*PointsPerC
ycle);
r=r0*exp(lambda*theta);
x[i]=r*cos(clockoranti*(theta+(2
.0*PI*Spiral)/(1.0*NumSpirals)))
;
y[i]=r*sin(clockoranti*(theta+(2
.0*PI*Spiral)/(1.0*NumSpirals)))
;
z[i]=0.0;
}

Colour=Layer;
for(i=1;i<=NoSegments;i+=1)DXFLi
nes();
}
}

Layer=1;
NoSegments=PointsPerCycle;
for(Circle=0;Circle<=2.0*NumberO
fCycles*NumSpirals;Circle+=1)
{
for(i=0;i<=NoSegments;i+=1)
{
theta=(2.0*PI*i)/(1.0*PointsPerC
ycle);
r=r0*exp((lambda*2.0*PI*Circle)/
(2.0*NumSpirals));
x[i]=r*cos(theta);
y[i]=r*sin(theta);
z[i]=0.0;
}

Colour=Layer;
for(i=1;i<=NoSegments;i+=1)DXFLi
nes();
}

Layer=1;
NoSegments=1;
for(MyLine=1;MyLine<=2.0*NumSpir
als;MyLine+=1)
{
theta=(2.0*PI*MyLine)/(2.0*NumSp
irals);
r=r0;
x[0]=r*cos(theta);
y[0]=r*sin(theta);
z[0]=0.0;

r=r0*exp(lambda*2.0*PI*NumberOfC
ycles);
x[1]=r*cos(theta);
y[1]=r*sin(theta);
z[1]=0.0;

Colour=Layer;
for(i=1;i<=NoSegments;i+=1)DXFLi
nes();
}

DXFFinishOff();
cout<<"DXF file written, end of
program\n";

return 0;
}

void DXFSetUp(void)
{
Emma<<"0\n";
Emma<<"SECTION\n";
Emma<<"2\n";
Emma<<"ENTITIES\n";
}

void DXFLines(void)
{
DXFx1=x[i-1];
DXFy1=y[i-1];
DXFz1=z[i-1];
DXFx2=x[i];
DXFy2=y[i];
DXFz2=z[i];

Emma<<"0\nLINE\n8\n"<<Layer<<"\n
";
Emma<<"10\n"<<DXFx1<<"\n";
Emma<<"20\n"<<DXFy1<<"\n";
Emma<<"30\n"<<DXFz1<<"\n";
Emma<<"11\n"<<DXFx2<<"\n";
Emma<<"21\n"<<DXFy2<<"\n";
Emma<<"31\n"<<DXFz2<<"\n";
Emma<<"62\n"<<Colour<<"\n";
}

void DXFFinishOff(void)
{
Emma<<"0\n";
Emma<<"ENDSEC\n";
Emma<<"0\n";
Emma<<"EOF\n";
Emma.close();
}
```

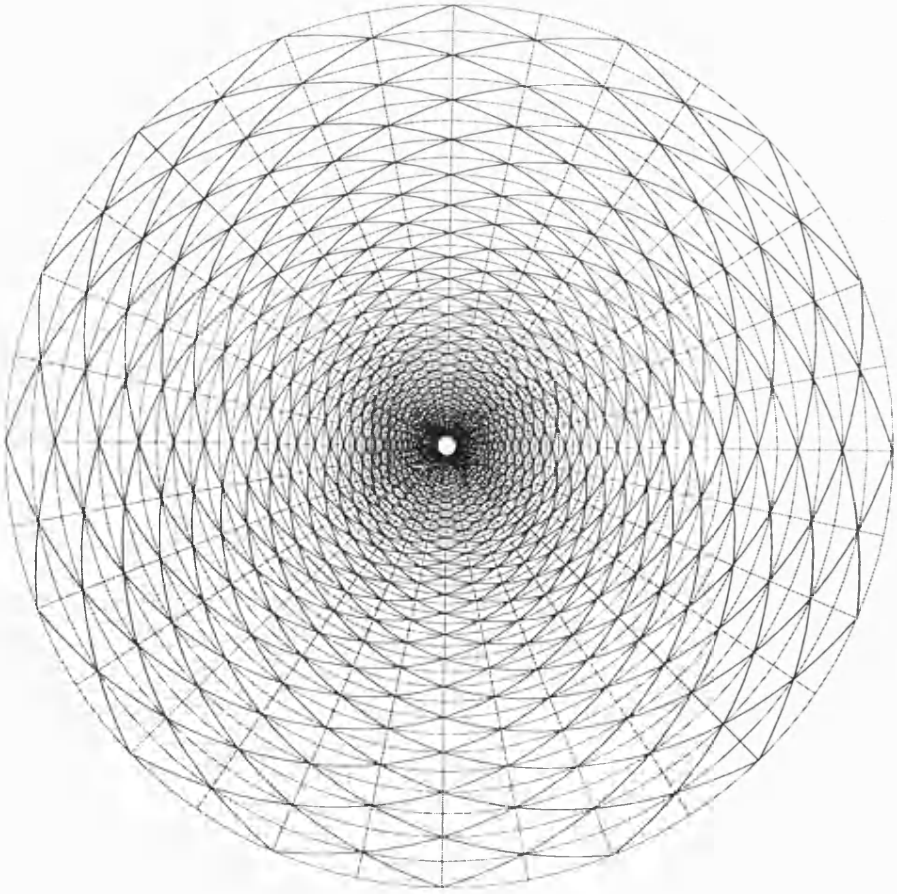


Figure A1.2a Multiple inter-locking spirals based on curved rectangles

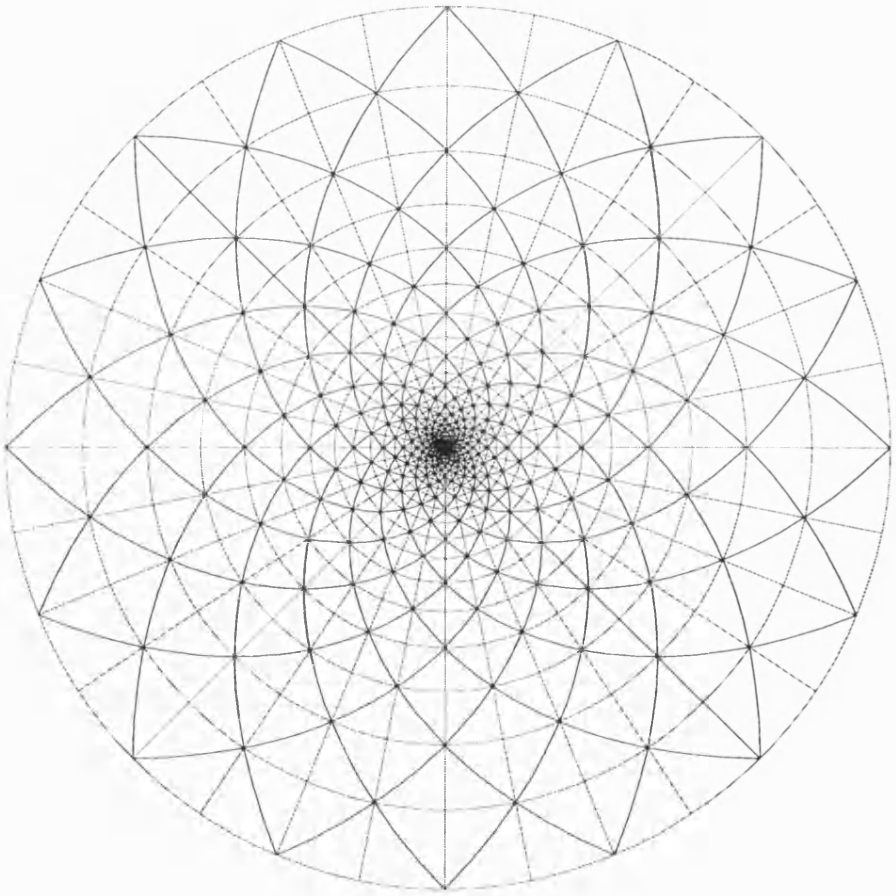


Figure A1.2b Multiple inter-locking spirals based on curved squares

A1.3 Computer Program To Generate a Shell

The following is a computer program written in C++, which incorporates a simple function describing the geometry of a shell. The result is shown in figure A1.3a as a transparent view, which was generated by rotating the circles along the axis of the spiral shown in figure A1.3b. As ϕ is varied in the program, one moves around the circle. Varying θ moves one from one circle to the next. The computer program produces a *dxf* file, which can be read into CAD and rendering programs.

Since this program also draws on the screen of a Macintosh computer, it will only run on a Macintosh. However, if those parts of the program are removed, it would run on a PC, or any other computer with a C++ compiler, producing a *dxf* file.

The terms used in the program are,

r = spiralradius; ρ = tuberadius and a = constant

and the relationships used in the program to generate the spiral correspond to

$$r = a\frac{e^{0.05\theta}}{e^{0.05\theta\max}},$$

and,

$$\rho = 0.9r,$$

and,

$$x = \rho\sin\phi - 4r + 3a$$

$$y = (r + \rho\cos\phi)\cos\theta$$

$$z = (r + \rho\cos\phi)\sin\theta.$$

```
// This is the start of the program; comments in red are not part of
the program

#include <math.h>
#include <fstream.h>
#include <iostream.h>
```

```
#define  mPlus1      301
#define  nPlus1       31
#define  halfW       317
#define  halfH       218
#define  downsh       40
#define  rightsh       2

double   x[mPlus1][nPlus1],y[mPlus1][nPlus1],z[mPlus1][nPlus1],
         PI,a,theta,thetamax,phi,spiralradius,tuberadius,
         scale,xplot[2],yplot[2];
int      i,j,m,n,Layer,intxscreen[2],intyscreen[2];

ofstream Emma("Shell.dxf");

WindowPtr   myWindow;
Rect        theRect;
RGBColor    Colour;

void dxfFace(void);
void OpenMacWindow(void);
void MacLine(void);
void CloseMacWindow(void);

int main(void)
{
PI=4.0*atan(1.0);a=10000.0;thetamax=10.0*PI;
scale=100.0;
// This next part of the program calculates x, y and z co-ordinates
m=mPlus1-1;n=nPlus1-1;
   for(j=0;j<=n;j+=1)
   {
   phi=(2.0*PI*j)/(1.0*n);
      for(i=0;i<=m;i+=1)
      {
      theta=(thetamax*i)/(1.0*m);
      spiralradius=a*exp(0.05*theta)/exp(0.05*thetamax);
      tuberadius=0.9*spiralradius;
      x[i][j]=tuberadius*sin(phi)-4.0*spiralradius+3.0*a;
      y[i][j]=(spiralradius+tuberadius*cos(phi))*cos(theta);
      z[i][j]=(spiralradius+tuberadius*cos(phi))*sin(theta);
      }
   }
// This part of the program draws on screen and to dxf file
OpenMacWindow();
Emma<<"0\nSECTION\n2\nENTITIES\n";
Layer=0;

   for(i=0;i<=m;i+=1)
   {
      for(j=0;j<=n;j+=1)
      {
      if(i<m&&j<n)dxfFace();
      xplot[0]=x[i][j];yplot[0]=y[i][j];
      if(i<m)
        {
        xplot[1]=x[i+1][j];yplot[1]=y[i+1][j];MacLine();
        }
      if(j<n)
        {
        xplot[1]=x[i][j+1];yplot[1]=y[i][j+1];MacLine();
        }
}
   }
Emma<<"0\nENDSEC\n0\nEOF\n";Emma.close();
CloseMacWindow();
```

```
cout<<"Finished, dxf file written.\n";
return 0;
}
// This part of the program writes to dxf file
void dxfFace(void)
{
      Emma<<"0\n3DFACE\n8\n"<<Layer<<"\n";
      Emma<<"10\n"<<x[i][j]<<"\n";
      Emma<<"20\n"<<y[i][j]<<"\n";
      Emma<<"30\n"<<z[i][j]<<"\n";
      Emma<<"11\n"<<x[i][j+1]<<"\n";
      Emma<<"21\n"<<y[i][j+1]<<"\n";
      Emma<<"31\n"<<z[i][j+1]<<"\n";
      Emma<<"12\n"<<x[i+1][j+1]<<"\n";
      Emma<<"22\n"<<y[i+1][j+1]<<"\n";
      Emma<<"32\n"<<z[i+1][j+1]<<"\n";
      Emma<<"13\n"<<x[i+1][j]<<"\n";
      Emma<<"23\n"<<y[i+1][j]<<"\n";
      Emma<<"33\n"<<z[i+1][j]<<"\n";
}
// This part of the program opens window on Macintosh
void OpenMacWindow(void)
{
   InitGraf(&qd.thePort);
   InitFonts();
   InitWindows();
   InitMenus();
   TEInit();
   InitDialogs(0L);
   FlushEvents(everyEvent,0);
   InitCursor();

   SetRect(&theRect,rightsh,downsh,rightsh+2*halfW,downsh+2*halfH);

   myWindow=NewCWindow(0L,&theRect,
   "\pCentre for Lightweight Structures, University of Bath, U.K.",
   true,documentProc,(WindowPtr)-1L,true,0L);
   SetPort(myWindow);
}

// This part of the program draws a line on the screen
void MacLine(void)
{
intxscreen[0]=halfW+xplot[0]/scale;
intxscreen[1]=halfW+xplot[1]/scale;

intyscreen[0]=halfH-yplot[0]/scale;
intyscreen[1]=halfH-yplot[1]/scale;

Colour.red=65535.0*1.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
RGBForeColor(&Colour);

MoveTo(intxscreen[0],intyscreen[0]);
LineTo(intxscreen[1],intyscreen[1]);
}
// This last part of the program closes Macintosh window
void CloseMacWindow(void)
{
while (!Button());
CloseWindow(myWindow);
}
```

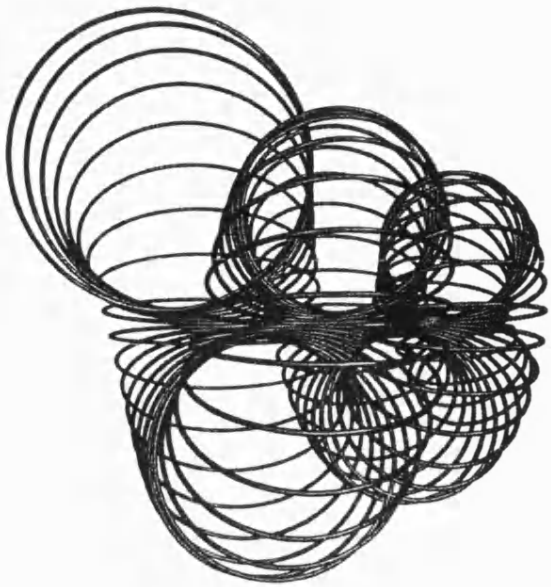


Figure A1.3a Shell constructed from rotated circles

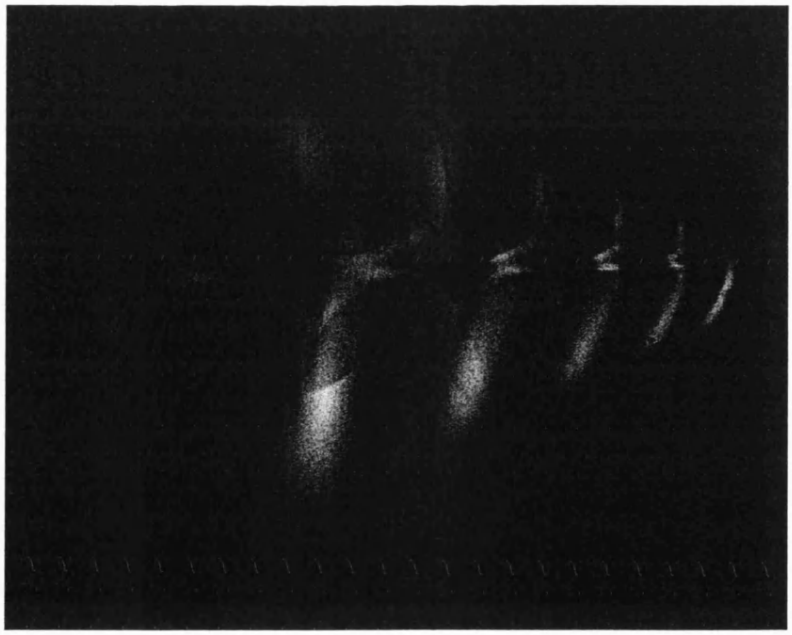


Figure A1.3b Transparent view of the shell

Appendix A2.0 Complex Analysis

A2.1 Complex Number Algebra

In normal algebra, $5\times 5=25$and $(-5)\times(-5)=25$. In complex number algebra $i=\sqrt{-1}$and $i\times i=-1$. In the following example $(5+7i)$ and $(4-2i)$ are added, subtracted, multiplied and divided.

$$(5+7i)+(4-2i)=(5+4)+(7-2)i=9+5i, \tag{1}$$

$$(5+7i)-(4-2i)=(5-4)+(7+2)i=1+9i, \tag{2}$$

$$\begin{aligned}(5+7i)(4-2i)&=5(4-2i)+7i(4-2i)\\&=20-10i+28i-14(i\times i)\\&=34+18i,\end{aligned} \tag{3}$$

and

$$\begin{aligned}\frac{5+7i}{4-2i}&=\frac{(5+7i)(4+2i)}{(4-2i)(4+2i)}\\&=\frac{5(4+2i)+7i(4+2i)}{4(4+2i)-2i(4+2i)}.\\&=\frac{20+10i+28i-14}{16+8i-8i+4}\\&=\frac{1}{20}(6+38i).\end{aligned} \tag{4}$$

If $z=x+iy$, then

$$\begin{aligned}z^2&=(x+iy)^2=x(x+iy)+iy(x+iy)\\&=(x^2-y^2)+2ixy\end{aligned} \tag{5}$$

and

$$\begin{aligned}\frac{1}{z} &= \frac{1}{x+iy} = \frac{x-iy}{(x+iy)(x-iy)} \\ &= \frac{x-iy}{x(x-iy)+iy(x-iy)} \\ &= \frac{x-iy}{x^2+y^2}.\end{aligned} \tag{6}$$

The exponential and trigonometric functions

$$e^{\theta} = 1+\theta+\frac{\theta^2}{2}+\frac{\theta^3}{3!}+\frac{\theta^4}{4!}+\frac{\theta^5}{5!}+\frac{\theta^6}{6!}+\ldots$$

$$\cos\theta = 1-\frac{\theta^2}{2}+\frac{\theta^4}{4!}-\frac{\theta^6}{6!}+\ldots \tag{7}$$

and

$$\sin\theta = \theta-\frac{\theta^3}{3!}+\frac{\theta^5}{5!}-\ldots \tag{8}$$

can be combined using complex numbers as follows,

$$\begin{aligned}e^{i\theta} &= 1+i\theta+\frac{(i\theta)^2}{2}+\frac{(i\theta)^3}{3!}+\frac{(i\theta)^4}{4!}+\frac{(i\theta)^5}{5!}+\frac{(i\theta)^6}{6!}+\ldots \\ &= 1+i\theta-\frac{\theta^2}{2}-i\frac{\theta^3}{3!}+\frac{\theta^4}{4!}+i\frac{\theta^5}{5!}-\frac{\theta^6}{6!}+\ldots \\ &= 1-\frac{\theta^2}{2}+\frac{\theta^4}{4!}-\frac{\theta^6}{6!}+\ldots i\left(\theta-\frac{\theta^3}{3!}+\frac{\theta^5}{5!}-\ldots\right) \\ &= \cos\theta+i\sin\theta.\end{aligned} \tag{9}$$

On the *Argand* diagram given in figure 3.2a

$x = r\cos\theta$ and $y = r\sin\theta$, the complex number $z = x+iy$ can be written

$$\begin{aligned}z = x+iy &= r(\cos\theta+i\sin\theta) \\ &= re^{i\theta}\end{aligned} \tag{10}$$

$$\begin{aligned} e^z &= e^{(x+iy)} = e^x e^{iy} \\ &= e^x(\cos y + i\sin y) \end{aligned} \tag{11}$$

The hyberbolic functions

$$\begin{aligned} \cosh z &= \frac{e^z + e^{-z}}{2} \\ &= \frac{e^x(\cos y + i\sin y) + e^{-x}(\cos(-y) + i\sin(-y))}{2} \\ &= \frac{e^x(\cos y + i\sin y) + e^{-x}(\cos y - i\sin y)}{2} \\ &= \cosh x\cos y + i\sinh x\sin y \end{aligned} \tag{12}$$

and

$$\begin{aligned} \sinh z &= \frac{e^z - e^{-z}}{2} \\ &= \frac{e^x(\cos y + i\sin y) - e^{-x}(\cos y - i\sin y)}{2} \\ &= \sinh x\cos y + i\cosh x\sin y. \end{aligned} \tag{13}$$

Also

$$\cosh(iz) = \frac{e^{iz} + e^{-iz}}{2} = \frac{(\cos z + i\sin z) + (\cos z - i\sin z)}{2} = \cos z \tag{14}$$

and

$$\sinh(iz) = \frac{e^{iz} - e^{-iz}}{2} = \frac{(\cos z + i\sin z) - (\cos z - i\sin z)}{2} = i\sin z. \tag{15}$$

The logarithm,

$$\log z = \log(re^{i\theta}) = \log r + i\theta. \tag{16}$$

$$z = x + iy \tag{17}$$

$$\eta = \phi + i\psi \tag{18}$$

Appendix A3.0 Conformal Maps

A3.1 Conformal Map 1

Below is the first example of a function that can be used to generate a simple conformal map. The computer program used to generate it is given in appendix A3.1.1. If $z = x + iy$ and $\eta = \phi + i\psi$, then if $z = \frac{a}{\eta}$,

$$z = \frac{a(\phi - i\psi)}{\phi^2 + \psi^2} \text{ which means that } x = \frac{a\phi}{\phi^2 + \psi^2} \text{ and } y = -\frac{a\psi}{\phi^2 + \psi^2} \text{ so that}$$

$$x^2 + y^2 = \frac{a^2}{\phi^2 + \psi^2}$$

and

$$ax = \phi(x^2 + y^2)$$

$$ay = -\psi(x^2 + y^2).$$

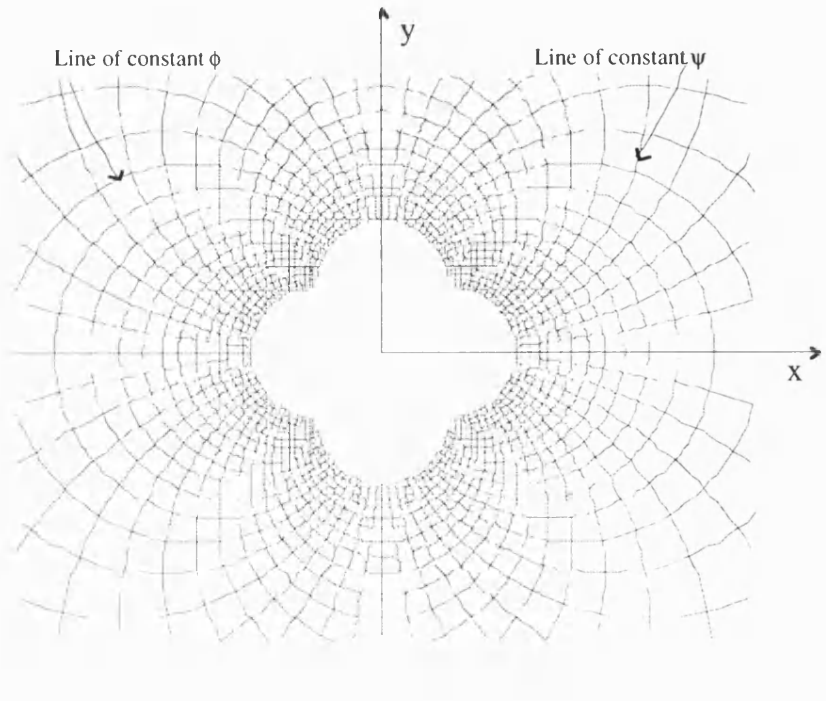


Figure A3.1 Conformal Map 1

A3.1.1 Computer Program to generate Conformal Map 1

```
#include <math.h>
#include <fstream.h>
#include <iostream.h>
#define  mPlus1       201
#define  nPlus1       201
float    x[mPlus1][nPlus1],y[mPlus1][nPlus1],z[mPlus1][nPlus1],
         PI,a,phi,psi,tempfloat;
int      i,j,m,n,step;
ofstream Emma("Map1.dxf");
int main(void)
{
PI=4.0*atan(1.0);a=5000.0;step=5;
m=(mPlus1-1)/step;m=m*step;
n=(nPlus1-1)/step;n=n*step;
   for(i=0;i<=m;i+=1)
   {
   phi=(1.0*i)/(1.0*m)-1.0/2.0;
      for(j=0;j<=n;j+=1)
      {
      psi=(1.0*j)/(1.0*m)-1.0/2.0;
      tempfloat=phi*phi+psi*psi;
      if(tempfloat>1.0e-6)
      {
      x[i][j]=a*phi/tempfloat;
      y[i][j]=-a*psi/tempfloat;
      }
      else
      {
      x[i][j]=0.0;
      y[i][j]=0.0;
      }
```

```
      z[i][j]=0.0;
      }
   }
cout<<"Please wait, writing file.\n";
Emma<<"0\rSECTION\r2\rENTITIES\r";
   for(i=0;i<=m;i+=step)
   {
      for(j=0;j<=n-1;j+=1)
      {

if((x[i][j]!=0.0||y[i][j]!=0.0)&&(x[i][j+1]!=0.0||y[i][j+1]!=0.0))
      {
      Emma<<"0\rLINE\r8\r0\r";
      Emma<<"10\r"<<x[i][j]<<"\r";
      Emma<<"20\r"<<y[i][j]<<"\r";
      Emma<<"30\r"<<z[i][j]<<"\r";
      Emma<<"11\r"<<x[i][j+1]<<"\r";
      Emma<<"21\r"<<y[i][j+1]<<"\r";
      Emma<<"31\r"<<z[i][j+1]<<"\r";
      }
      }
   }
   for(j=0;j<=n;j+=step)
   {
      for(i=0;i<=m-1;i+=1)
      {

if((x[i][j]!=0.0||y[i][j]!=0.0)&&(x[i+1][j]!=0.0||y[+1][j]!=0.0))
      {
      Emma<<"0\rLINE\r8\r0\r";
      Emma<<"10\r"<<x[i][j]<<"\r";
      Emma<<"20\r"<<y[i][j]<<"\r";
      Emma<<"30\r"<<z[i][j]<<"\r";
      Emma<<"11\r"<<x[i+1][j]<<"\r";
      Emma<<"21\r"<<y[i+1][j]<<"\r";
      Emma<<"31\r"<<z[i+1][j]<<"\r";
      }
      }
   }
Emma<<"0\rENDSEC\r0\rEOF\r";Emma.close();
cout<<"Finished, dxf file written.\n";
return 0;
}
```

A3.2 Conformal Map 2

The computer program used to generate this map is given in appendix A3.2.1

$$z = a\sqrt{\eta}$$

$$a^2\eta = z^2 = x^2 - y^2 + 2ixy$$

$$a^2\phi = x^2 - y^2 \text{ and } a^2\psi = 2xy$$

$$a^2\phi = x^2 - \left(\frac{\psi}{2x}\right)^2 \text{ or } x^4 - a^2\phi x^2 - a^4\left(\frac{\psi}{2}\right)^2 = 0$$

$$\frac{x^2}{a^2} = \frac{\phi \pm \sqrt{\phi^2 + \psi^2}}{2}$$

$$\frac{x}{a} = \frac{1}{\sqrt{2}}\sqrt{\phi + \sqrt{\phi^2 + \psi^2}} = \frac{1}{\sqrt{2}}\frac{\psi}{\sqrt{-\phi + \sqrt{\phi^2 + \psi^2}}}$$

$$\frac{y}{a} = \frac{1}{\sqrt{2}}\frac{\psi}{\sqrt{\phi + \sqrt{\phi^2 + \psi^2}}} = \frac{1}{\sqrt{2}}\sqrt{-\phi + \sqrt{\phi^2 + \psi^2}}$$

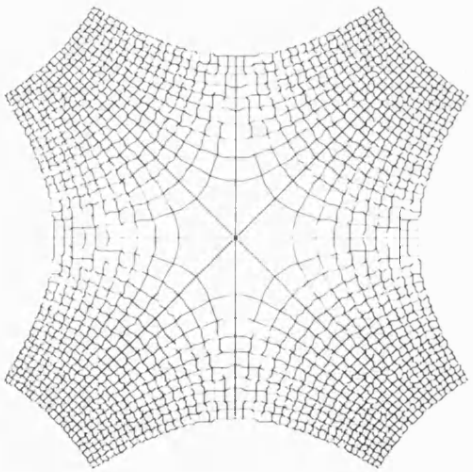


Figure A3.2 Conformal Map 2

A3.2.1 Computer Program to generate Conformal Map 2

```
#include <math.h>
#include <fstream.h>
#include <iostream.h>
#define  mPlus1       61
#define  nPlus1       61
float    x[mPlus1][nPlus1],y[mPlus1][nPlus1],z[mPlus1][nPlus1],
         PI,a,phi,psi,tempfloat;
int      i,j,m,n,step,mirrorx,mirrory;
ofstream Emma("Map2.dxf");
int main(void)
{
PI=4.0*atan(1.0);a=50000.0;step=3;
m=(mPlus1-1)/step;m=m*step;
n=(nPlus1-1)/step;n=n*step;
   for(i=0;i<=m;i+=1)
   {
```

```
   phi=(1.0*i)/(1.0*m)-0.5;
      for(j=0;j<=n;j+=1)
      {
      psi=(1.0*j)/(1.0*m);
      tempfloat=sqrt(phi*phi+psi*psi);
      x[i][j]=a*sqrt(+phi+tempfloat)/sqrt(2.0);
      y[i][j]=a*sqrt(-phi+tempfloat)/sqrt(2.0);
      z[i][j]=0.0;
      }
   }
cout<<"Please wait, writing file.\n";
Emma<<"0\rSECTION\r2\rENTITIES\r";
for(mirrorx=-1;mirrorx<=1;mirrorx+=2)
{
for(mirrory=-1;mirrory<=1;mirrory+=2)
{
   for(i=0;i<=m;i+=step)
   {
      for(j=0;j<=n-1;j+=1)
      {
      Emma<<"0\rLINE\r8\r0\r";
      Emma<<"10\r"<<mirrorx*x[i][j]<<"\r";
      Emma<<"20\r"<<mirrory*y[i][j]<<"\r";
      Emma<<"30\r"<<z[i][j]<<"\r";
      Emma<<"11\r"<<mirrorx*x[i][j+1]<<"\r";
      Emma<<"21\r"<<mirrory*y[i][j+1]<<"\r";
      Emma<<"31\r"<<z[i][j+1]<<"\r";
      }
   }
   for(j=0;j<=n;j+=step)
   {
      for(i=0;i<=m-1;i+=1)
      {
      Emma<<"0\rLINE\r8\r0\r";
      Emma<<"10\r"<<mirrorx*x[i][j]<<"\r";
      Emma<<"20\r"<<mirrory*y[i][j]<<"\r";
      Emma<<"30\r"<<z[i][j]<<"\r";
      Emma<<"11\r"<<mirrorx*x[i+1][j]<<"\r";
      Emma<<"21\r"<<mirrory*y[i+1][j]<<"\r";
      Emma<<"31\r"<<z[i+1][j]<<"\r";
      }
   }
}
}
Emma<<"0\rENDSEC\r0\rEOF\r";Emma.close();
cout<<"Finished, dxf file written.\n";
return 0;
}
```

A3.3 Conformal Map 3

The computer program used to generate this map follows in appendix A3.3.1 after the relationships given below.

$$\frac{z}{a} = \cosh\eta = \cosh\phi\cos\psi + i\sinh\phi\sin\psi$$

$$\text{so that } \frac{x}{a} = \cosh\phi\cos\psi \text{ and } \frac{y}{a} = \sinh\phi\sin\psi$$

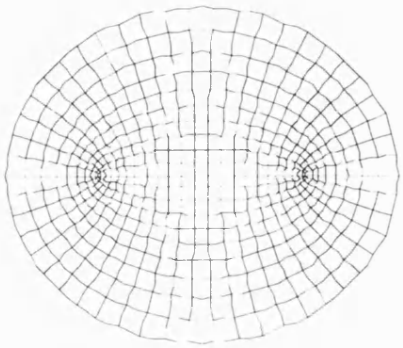


Figure A3.3 Conformal Map 3

A3.3.1 Computer Program to generate Conformal Map 3

```
#include <math.h>
#include <fstream.h>
#include <iostream.h>
#define  mPlus1        21
#define  nPlus1       101
float    x[mPlus1][nPlus1],y[mPlus1][nPlus1],z[mPlus1][nPlus1],
         PI,a,phi,psi,tempfloat;
int      i,j,m,n,step;
ofstream Emma("Map3.dxf");
int main(void)
{
PI=4.0*atan(1.0);a=20000.0;step=2;
m=(mPlus1-1)/step;m=m*step;
n=(nPlus1-1)/step;n=n*step;
   for(i=0;i<=m;i+=1)
   {
   phi=2.0*PI*(1.0*i)/(1.0*n);
      for(j=0;j<=n;j+=1)
      {
      psi=2.0*PI*(1.0*j)/(1.0*n);
      x[i][j]=a*cosh(phi)*cos(psi);
      y[i][j]=a*sinh(phi)*sin(psi);
      z[i][j]=0.0;
      }
   }
cout<<"Please wait, writing file.\n";
Emma<<"0\rSECTION\r2\rENTITIES\r";
   for(i=0;i<=m;i+=step)
   {
      for(j=0;j<=n-1;j+=1)
      {
      Emma<<"0\rLINE\r8\r0\r";
      Emma<<"10\r"<<x[i][j]<<"\r";
      Emma<<"20\r"<<y[i][j]<<"\r";
      Emma<<"30\r"<<z[i][j]<<"\r";
```

```
      Emma<<"11\r"<<x[i][j+1]<<"\r";
      Emma<<"21\r"<<y[i][j+1]<<"\r";
      Emma<<"31\r"<<z[i][j+1]<<"\r";
      }
   }
   for(j=0;j<=n;j+=step)
   {
      for(i=0;i<=m-1;i+=1)
      {
      Emma<<"0\rLINE\r8\r0\r";
      Emma<<"10\r"<<x[i][j]<<"\r";
      Emma<<"20\r"<<y[i][j]<<"\r";
      Emma<<"30\r"<<z[i][j]<<"\r";
      Emma<<"11\r"<<x[i+1][j]<<"\r";
      Emma<<"21\r"<<y[i+1][j]<<"\r";
      Emma<<"31\r"<<z[i+1][j]<<"\r";
      }
   }
Emma<<"0\rENDSEC\r0\rEOF\r";Emma.close();
cout<<"Finished, dxf file written.\n";
return 0;
}
```

Appendix A4.0 *Dxf* File content and structure

A4.1 Standard *dxf* Group Codes and Value Types

Group Code Value Type:

0	Group code 0 identifies the start of a section. In this case, it defines the start of the entities section. It also acts as a file separator.
2	Group code 2 is for the name of a Section. In the example below, the entities section has two types of data. The first type of data consists of lines, and the third type of data consists of 3Dfaces.
8	Group code 8 is reserved for the Layer name on which an entity appears. In the example below, the Ring co-ordinates appear on the Rings layer, the Hoop co-ordinates appear on the Hoops layer and the Surface co-ordinates appear on the Surface layer.
10	Primary *x* co-ordinate
11 – 18	Other *x* co-ordinates
20	Primary *y* co-ordinate
21 – 28	Other *y* co-ordinates
30	Primary *z* co-ordinate
31 – 37	Other *z* co-ordinates

Format of a typical Entities Section:

0	(Beginning of a section)
SECTION	
2	(Name of Section follows)
ENTITIES	(Drawing entities appear here)
0	(File separator)
ENDSEC	(End of section)
EOF	(End of file)

A4.2 Example of a C++ instruction to create *dxf* output file:

```
Rest<<"0\rSECTION\r2\rENTITIES\r";
for(i=0;i<=m-step;i+=step)
```

```
{
for(j=0;j<=n-1;j+=1)
{
Rest<<"0\rLINE\r8\rRings\r";
            Rest<<"10\r"<<x[i][j]<<"\r";
            Rest<<"20\r"<<y[i][j]<<"\r";
            Rest<<"30\r"<<z[i][j]<<"\r";
            Rest<<"11\r"<<x[i][j+1]<<"\r";
            Rest<<"21\r"<<y[i][j+1]<<"\r";
      Rest<<"31\r"<<z[i][j+1]<<"\r";
}
for(j=0;j<=n-step;j+=step)
{
for(i=0;i<=m-1;i+=1)
     {
            Rest<<"0\rLINE\r8\rHoops\r";
            Rest<<"10\r"<<x[i][j]<<"\r";
            Rest<<"20\r"<<y[i][j]<<"\r";
            Rest<<"30\r"<<z[i][j]<<"\r";
            Rest<<"11\r"<<x[i+1][j]<<"\r";
            Rest<<"21\r"<<y[i+1][j]<<"\r";
            Rest<<"31\r"<<z[i+1][j]<<"\r";
      }
}
for(i=0;i<=m-1;i+=1)
{
for(j=0;j<=n-1;j+=1)
     {
Rest<<"0\r3DFACE\r8\rSurface\r";
            Rest<<"10\r"<<x[i][j]<<"\r";
            Rest<<"20\r"<<y[i][j]<<"\r";
            Rest<<"30\r"<<z[i][j]<<"\r";
            Rest<<"11\r"<<x[i][j+1]<<"\r";
            Rest<<"21\r"<<y[i][j+1]<<"\r";
      Rest<<"31\r"<<z[i][j+1]<<"\r";
            Rest<<"12\r"<<x[i+1][j+1]<<"\r";
            Rest<<"22\r"<<y[i+1][j+1]<<"\r";
            Rest<<"32\r"<<z[i+1][j+1]<<"\r";
            Rest<<"13\r"<<x[i+1][j]<<"\r";
            Rest<<"23\r"<<y[i+1][j]<<"\r";
            Rest<<"33\r"<<z[i+1][j]<<"\r";
      }
}
Rest<<"0\rENDSEC\r0\rEOF\r";Rest.close();
Return 0;
}
```

A4.3 Example of *dxf* file created by C++ instruction:

The text in italics is for explanatory purposes only, and does not form a part of the output file.

0 (*Start of an entry or Section; Code 0 also acts as a file separator*)

SECTION

2 (*This group code is for naming the type of dxf Section*)

ENTITIES

0 (*Start of entity, file separator*)

LINE (*Name of entity type)*

8 (*Layer name of entity follows*)

Rings

10 (*Primary x co-ordinate follows indicating the start of line*)

18000 (*Actual value of x co-ordinate*)

20 (*Primary y co-ordinate follows indicating the start of a line*)

0 (*Actual value of y co-ordinate*)

30 (*Primary z co-ordinate follows indicating the start of a line*)

0 (*Actual value of z co-ordinate*)

11 (*Other x co-ordinate follows indicating the end point of a line*)

17952.7

21 (*Other y co-ordinate follows indicating the end point of a line*)

751.999

31 (*Other z co-ordinate follows indicating the end point of a line*)

0

0 (*File separator*)

LINE

8

Rings

10

17952.7

20

751.999

30

0

11

17811.5

21

1492.14

31

0

0

This process continues until the co-ordinates of all lines are listed. The complete list is often several or even hundreds pages long and therefore has not been included in this work in its entirety.

It will be observed from the above that the end point of a given line shares the same co-ordinates as the start point of the next line. This is necessary to achieve continuity of the shape described.

The next section is a continuation of the above list, and this time the co-ordinates give the four corners of a 3D face.

3DFACE (*Name of entity type*)

8 (*Layer name of entity follows*)

Surface

10 (*Primary x co-ordinate follows indicating the first corner of a 3D face*)

17776.4 (*Actual value of x co-ordinate*)

20 (*Primary y co-ordinate follows indicating the first corner of a 3D face*)

-1492.14 (*Actual value of y co-ordinate*)

30 (*Primary z co-ordinate follows indicating the first corner of a 3D face*)

-1118.39 (*Actual value of z co-ordinate*)

11 (*Other x co-ordinate follows indicating the second corner of a 3D face*)

17917.3

21 (*Other y co-ordinate follows indicating the second corner of a 3D face*)

-751.999

31 (*Other z co-ordinate follows indicating the second corner of a 3D face*)

-1127.26

12 (*Other x co-ordinate follows indicating the third corner of a 3D face*)

17952.7

22 (*Other y co-ordinate follows indicating the third corner of a 3D face*)

-751.999

32 (*Other z co-ordinate follows indicating the third corner of a 3D face*)

0

13 (*Other x co-ordinate follows indicating the fourth corner of a 3D face*)

17811.5

23 (*Other y co-ordinate follows indicating the fourth corner of a 3D face*)

-1492.14

33 (*Other z co-ordinate follows indicating the fourth corner of a 3D face*)

0

0 (*File separator*)

ENDSEC (*Dxf file must indicate end of a section thus*)

0 (*File separator*)

EOF (*Indicator for end of file; every dxf file must have an end of file*)

Appendix B Branching Study Computer Programs

B1.0 Road Bridge Study 2 computer program

```
#include <math.h>
#include <stdio.h>

#define    msurfnum      201
#define    nsurfnum       51
#define    sourcenum     11

#define    dxf            1

#define    downsh        40
#define    rightsh        2

void findxy(void);
void potential(void);

void SetUpDrawingArea(void);
void DrawPicy(void);

void dxfSetUp(void);
void dxfmakeobject(void);
void dxfFinishOff(void);

double xcalc[msurfnum][nsurfnum][2],ycalc[msurfnum][nsurfnum][2],

xplot[msurfnum][nsurfnum],yplot[msurfnum][nsurfnum],zplot[msurfnum][ns
urfnum],
       PI,scale,phi,psi,x,y,
       a[sourcenum],b[sourcenum],c[sourcenum],
       tempfloat,real,imag,deltax,deltay,deltaphi,deltapsi,
       phiContrib,psiContrib,
       realdiffContrib,imagdiffContrib,
       realdiff,imagdiff,
       halfspan,strength[sourcenum],strengthsofar,totalstrength,
       theta,mirror1,mirror2,mirror3,Fourier,maxwidth[msurfnum],
       bflat,cflat,
       tuberad,ytube1,ztube1,ytube2,ztube2,facetangle;

int    i,j,m,n[sourcenum],halfW,halfH,
      intxscreen[2],intyscreen[2],
      counter,
      numsourcesminus1,source,
      whichbit,upperorlowerbit,nF,
      whichview,numberofwaves,
      smooth,facet,nfacets,diagonal,step,
      ii,jj,jincrement,jstart,jstop;

WindowPtr   myWindow;
Rect        theRect;
RGBColor    Colour;


FILE *Picy;

void main(void)
{
step=6;

printf("Smooth surface = 1\n");
```

```
scanf("%d",&smooth);

PI=4.0*atan(1.0);
scale=300.0;
halfspan=80000.0;

//The following must be an even multiple times step
n[2]=2*step;
n[1]=2*step;
n[0]=2*step;

m=80;
nfacets=6;

numberofwaves=4;

numsourcesminus1=2;

bflat=6000.0;
cflat=200.0;

a[2]=0.0;
c[2]=10000.0;

a[1]=0.0;
c[1]=10000.0;

a[0]=0.0;
c[0]=10000.0;

SetUpDrawingArea();
if(dxf==1)dxfSetUp();

for(whichbit=numsourcesminus1;whichbit>=0;whichbit-=1)
{
for(upperorlowerbit=0;upperorlowerbit<=1;upperorlowerbit+=1)
{
for(i=0;i<=m;i+=1)
{
theta=(PI*i)/(1.0*m);

strength[2]=5.0;
strength[1]=2.0;//*(5.0+cos(theta))/6.0;
strength[0]=4.0;//*(8.0+cos(theta))/9.0;

totalstrength=0.0;
for(source=0;source<=numsourcesminus1;source+=1)totalstrength+=strengt
h[source];
strengthsofar=0.0;
for(source=numsourcesminus1;source>=whichbit;source-
=1)strengthsofar+=strength[source];

if(whichbit==numsourcesminus1&&upperorlowerbit==0)maxwidth[i]=0.0;

Fourier=-1/3.0;
for(nF=1;nF<=5;nF+=1)
{
tempfloat=1.0*nF;
Fourier-=cos(nF*theta)/(tempfloat*tempfloat);
}
Fourier=Fourier*4.0/(PI*PI);
b[2]=6000.0;
b[0]=b[2]-4500.0+16000.0*Fourier;

Fourier=0.0;
```

```
for(nF=1;nF<=5;nF+=1)
{
tempfloat=2.0*nF-1.0;
Fourier+=cos(numberofwaves*tempfloat*theta)/(tempfloat*tempfloat);
}
Fourier=Fourier*8.0/(PI*PI);
b[1]=(b[2]+b[0])/2.0-(b[2]-b[0])*Fourier/2.0;

phi=-3.0*PI-6.0*PI*(1.0-cos(theta))/2.0;
if(i!=0)
{
x=xcalc[i-1][0][upperorlowerbit];
y=ycalc[i-1][0][upperorlowerbit];
}
else{x=0.6*c[whichbit];y=b[whichbit];}
for(j=0;j<=n[whichbit];j+=1)
{
if(smooth==1)
psi=(1.0*totalstrength-
2.0*strengthsofar+1.0*strength[whichbit])*PI/2.0
        +sin((PI*j)/(2.0*n[whichbit]))*((2.0*upperorlowerbit-
1.0)*PI*strength[whichbit])/2.0;
else
psi=(1.0*totalstrength-
2.0*strengthsofar+1.0*strength[whichbit])*PI/2.0
          +((1.0*j)/(1.0*n[whichbit]))*((2.0*upperorlowerbit-
1.0)*PI*strength[whichbit])/2.0;
findxy();
xcalc[i][j][upperorlowerbit]=x;
ycalc[i][j][upperorlowerbit]=y;
if(whichbit==numsourcesminus1&&maxwidth[i]<fabs(x))maxwidth[i]=fabs(x)
;
}
}
}

for(upperorlowerbit=0;upperorlowerbit<=1;upperorlowerbit+=1)
{
for(i=0;i<=m;i+=1)
{
for(j=0;j<=n[whichbit];j+=1)
{
yplot[i][j]=xcalc[i][j][upperorlowerbit]*2000.0/maxwidth[i];
zplot[i][j]=ycalc[i][j][upperorlowerbit];
zplot[i][j]=(zplot[i][j]+bflat-sqrt((zplot[i][j]-bflat)*(zplot[i][j]-
bflat)+4.0*cflat*cflat))/2.0;
xplot[i][j]=(halfspan*i)/(1.0*m);
}
}

DrawPicy();
if(dxf==1)dxfmakeobject();
}
}

if(dxf==1)dxfFinishOff();
while (!Button());
CloseWindow(myWindow);
}

void findxy(void)
{
for(counter=1;counter<=100;counter+=1)
{
potential();
```

```
x+=deltax;
y+=deltay;

if(deltax*deltax+deltay*deltay<1.0e-8)break;
if(x<0.0)x=0.0;
}
}

void potential(void)
{
deltaphi=phi;
deltapsi=psi;

realdiff=0.0;
imagdiff=0.0;

for(source=0;source<=numsourcesminus1;source+=1)
{
real=(x-a[source])/c[source];
imag=(y-b[source])/c[source];
tempfloat=sqrt(real*real+imag*imag);
phiContrib=log(tempfloat);
if(fabs(real)>fabs(imag))
{
psiContrib=asin(imag/tempfloat);
if(real<0.0&&imag>=0.0)psiContrib=PI-psiContrib;
if(real<0.0&&imag<0.0)psiContrib=-PI-psiContrib;
}
else
{
psiContrib=acos(real/tempfloat);
if(imag<0.0)psiContrib=-psiContrib;
}

tempfloat=c[source]*(real*real+imag*imag);
realdiffContrib=real/tempfloat;
imagdiffContrib=-imag/tempfloat;

deltaphi-=strength[source]*phiContrib;
deltapsi-=strength[source]*psiContrib;

realdiff+=strength[source]*realdiffContrib;
imagdiff+=strength[source]*imagdiffContrib;
}

tempfloat=realdiff*realdiff+imagdiff*imagdiff;

deltax=(deltaphi*realdiff+deltapsi*imagdiff)/tempfloat;
deltay=(deltapsi*realdiff-deltaphi*imagdiff)/tempfloat;
}

void dxfSetUp(void)
{
Picy=fopen("Form.dxf","w");
fprintf(Picy,"0\n");
fprintf(Picy,"SECTION\n");
fprintf(Picy,"2\n");
fprintf(Picy,"ENTITIES\n");
}

void dxfmakeobject(void)
{
for(mirror3=-1;mirror3<=1;mirror3+=2)
{
```

```
for(mirror2=-1;mirror2<=1;mirror2+=2)
{
for(mirror1=-1;mirror1<=1;mirror1+=2)
{
if(smooth==1)
{
for(i=0;i<=m-1;i+=1)
{
for(j=0;j<=n[whichbit]-1;j+=1)
{
fprintf(Picy,"0\n3DFACE\n8\n0\n");
fprintf(Picy,"10\n%f\n",mirror2*xplot[i][j]+mirror3*halfspan);
fprintf(Picy,"20\n%f\n",mirror1*yplot[i][j]);
fprintf(Picy,"30\n%f\n",zplot[i][j]);
fprintf(Picy,"11\n%f\n",mirror2*xplot[i][j+1]+mirror3*halfspan);
fprintf(Picy,"21\n%f\n",mirror1*yplot[i][j+1]);
fprintf(Picy,"31\n%f\n",zplot[i][j+1]);
fprintf(Picy,"12\n%f\n",mirror2*xplot[i+1][j+1]+mirror3*halfspan);
fprintf(Picy,"22\n%f\n",mirror1*yplot[i+1][j+1]);
fprintf(Picy,"32\n%f\n",zplot[i+1][j+1]);
fprintf(Picy,"13\n%f\n",mirror2*xplot[i+1][j]+mirror3*halfspan);
fprintf(Picy,"23\n%f\n",mirror1*yplot[i+1][j]);
fprintf(Picy,"33\n%f\n",zplot[i+1][j]);
}
}
}
}
else
{
for(diagonal=0;diagonal<=2;diagonal+=1)
{
if(diagonal==0)tuberad=100.0;
else tuberad=50.0;

if(diagonal==0)jincrement=step;
if(diagonal==1)jincrement=2*step;
if(diagonal==2)jincrement=-2*step;

if(diagonal==0)jstart=0;
if(diagonal==1)jstart=-1;
if(diagonal==2)jstart=n[whichbit]+1;

if(diagonal==0)jstop=n[whichbit];

for(i=0;i<=m-1;i+=1)
{
if(diagonal==0)jstart=0;
if(diagonal==1){jstart+=1;if(jstart>=2*step)jstart=0;}
if(diagonal==2){jstart-=1;if(jstart<=n[whichbit]-
2*step)jstart=n[whichbit];}

if(diagonal==0)jstop=n[whichbit];
if(diagonal==1)jstop=n[whichbit]-1;
if(diagonal==2)jstop=1;

for(j=jstart;(diagonal!=2&&j<=jstop)||(diagonal==2&&j>=jstop);j+=jincr
ement)
{
ytube1=tuberad;
ztube1=0.0;
ii=i+1;

if(diagonal==0)jj=j;
if(diagonal==1)jj=j+1;
if(diagonal==2)jj=j-1;
```

```
for(facet=1;facet<nfacets;facet+=1)
{
facetangle=(2.0*PI*facet)/(1.0*nfacets);
ytube2=tuberad*cos(facetangle);
ztube2=tuberad*sin(facetangle);
fprintf(Picy,"0\n3DFACE\n8\n0\n");
fprintf(Picy,"10\n%f\n",mirror2*xplot[i][j]+mirror3*halfspan);
fprintf(Picy,"20\n%f\n",mirror1*yplot[i][j]+ytube1);
fprintf(Picy,"30\n%f\n",zplot[i][j]+ztube1);
fprintf(Picy,"11\n%f\n",mirror2*xplot[ii][jj]+mirror3*halfspan);
fprintf(Picy,"21\n%f\n",mirror1*yplot[ii][jj]+ytube1);
fprintf(Picy,"31\n%f\n",zplot[ii][jj]+ztube1);
fprintf(Picy,"12\n%f\n",mirror2*xplot[ii][jj]+mirror3*halfspan);
fprintf(Picy,"22\n%f\n",mirror1*yplot[ii][jj]+ytube2);
fprintf(Picy,"32\n%f\n",zplot[ii][jj]+ztube2);
fprintf(Picy,"13\n%f\n",mirror2*xplot[i][j]+mirror3*halfspan);
fprintf(Picy,"23\n%f\n",mirror1*yplot[i][j]+ytube2);
fprintf(Picy,"33\n%f\n",zplot[i][j]+ztube2);
ytube1=ytube2;
ztube1=ztube2;
}
}
}
}
}
}
}
}
}

void dxfFinishOff(void)
{
fprintf(Picy,"0\n");
fprintf(Picy,"ENDSEC\n");
fprintf(Picy,"0\n");
fprintf(Picy,"EOF\n");
fclose(Picy);
}

void SetUpDrawingArea(void)
{
   InitGraf(&qd.thePort);
   InitFonts();
   InitWindows();
   InitMenus();
   TEInit();
   InitDialogs(0L);
   FlushEvents(everyEvent,0);
   InitCursor();

   halfW=317;halfH=218;

   SetRect(&theRect,rightsh,downsh,rightsh+2*halfW,downsh+2*halfH);

   myWindow=NewCWindow(0L,&theRect,
   "\pCentre for Lightweight Structures, University of Bath, U.K.",
   true,documentProc,(WindowPtr)-1L,true,0L);
   SetPort(myWindow);
}

void DrawPicy(void)
{
for(whichview=0;whichview<=1;whichview+=1)
{
if(upperorlowerbit==1)
```

```
{
Colour.red=65535.0*1.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
}
else
{
Colour.red=65535.0*0.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
}
RGBForeColor(&Colour);
for(j=0;j<=n[whichbit];j+=1)
{
for(i=0;i<=m-1;i+=1)
{
if(whichview==0)
{
intxscreen[0]=xplot[i][j]/scale;
intxscreen[1]=xplot[i+1][j]/scale;
intyscreen[0]=yplot[i][j]/scale;
intyscreen[1]=yplot[i+1][j]/scale;
}
else
{
intxscreen[0]=-xplot[i][j]/scale;
intxscreen[1]=-xplot[i+1][j]/scale;
intyscreen[0]=zplot[i][j]/scale;
intyscreen[1]=zplot[i+1][j]/scale;
}
MoveTo(halfW+intxscreen[0],halfH-intyscreen[0]);
LineTo(halfW+intxscreen[1],halfH-intyscreen[1]);
}
}

for(i=0;i<=m;i+=1)
{
for(j=0;j<=n[whichbit]-1;j+=1)
{
if(whichview==0)
{
intxscreen[0]=xplot[i][j]/scale;
intxscreen[1]=xplot[i][j+1]/scale;
intyscreen[0]=yplot[i][j]/scale;
intyscreen[1]=yplot[i][j+1]/scale;
}
else
{
intxscreen[0]=-xplot[i][j]/scale;
intxscreen[1]=-xplot[i][j+1]/scale;
intyscreen[0]=zplot[i][j]/scale;
intyscreen[1]=zplot[i][j+1]/scale;
}
MoveTo(halfW+intxscreen[0],halfH-intyscreen[0]);
LineTo(halfW+intxscreen[1],halfH-intyscreen[1]);
}
}
}
}
```

B2.0 Footbridge Study computer program: Calculates co-ordinates of section and elevation maps

```
#include <math.h>
#include <stdio.h>
```

```
#define    tol   1.0e-6
#define    bigtol   1.0e-6

void NewBranching(float geometry[],float complex[]);
void expthingy(float values[],float complex[],float oldcomplex[]);

void cosm1exp(float complex[]);
void squareroot(void);
void recttopolar(void);

float rad,angle,real,imag;

extern void NewBranching(float geometry[],float complex[])
{
float a,b,theta,psi,Re,Im,tempfloat;

a=geometry[0];
b=geometry[1];

theta=complex[0];
psi=complex[1];

 tempfloat=sinh(theta)*sinh(theta)*cos(psi)*cos(psi);
tempfloat+=cosh(theta)*cosh(theta)*sin(psi)*sin(psi);

Re=cosh(theta)*sinh(theta)/tempfloat;
Im=-cos(psi)*sin(psi)/tempfloat;

real=Re*Re-Im*Im;
imag=2.0*Re*Im;

real-=4.0*a*b/((a+b)*(a+b));

squareroot();

real=-Re-real;
imag=-Im-imag;

complex[0]=-(a+b)*imag/2.0;
complex[1]=-(a+b)*(1.0+real)/2.0;//Origin is moved and thing inverted
}

extern    void    expthingy(float    values[],float    complex[],float
oldcomplex[])
{
float Re,Im,really,imagy,
      x,y,theta,psi,Redzbydphi,Imdzbydphi,
      alpha,L,tempfloat,realerror,imagerror,realmov,imagmov,maxmov,PI;

int cycle;

PI=4.0*atan(1.0);

alpha=values[0];
L=values[1];

maxmov=L/100.0;
x=oldcomplex[0];
y=oldcomplex[1];

//if(alpha==0.0)
//{
//cosm1exp(complex);
//complex[0]=L*complex[0]/PI;
//complex[1]=L*complex[1]/PI;
```

```
//}
//else
//{
cycle=0;

again:
cycle+=1;

real=-(alpha+1.0)*PI*y/L;
imag=(alpha+1.0)*PI*x/L;

tempfloat=exp(real);

Re=tempfloat*cos(imag);
Im=tempfloat*sin(imag);

real=-(alpha-1.0)*PI*y/L;
imag=(alpha-1.0)*PI*x/L;

tempfloat=exp(real);

Re+=tempfloat*cos(imag);
Im+=tempfloat*sin(imag);

real=Re/2.0;
imag=Im/2.0;

recttopolar();

theta=2.0*PI-angle;
if(theta>PI)theta-=2.0*PI;
psi=log(rad);

realerror=complex[0]-theta;
imagerror=complex[1]-psi;

real=-PI*y/L;
imag=PI*x/L;

tempfloat=exp(real);

Re=(alpha+1.0)*tempfloat*cos(imag);
Im=(alpha+1.0)*tempfloat*sin(imag);
really=tempfloat*cos(imag);
imagy=tempfloat*sin(imag);

real=PI*y/L;
imag=-PI*x/L;

tempfloat=exp(real);

Re+=(alpha-1.0)*tempfloat*cos(imag);
Im+=(alpha-1.0)*tempfloat*sin(imag);
really+=tempfloat*cos(imag);
imagy+=tempfloat*sin(imag);

tempfloat=Re*Re+Im*Im;

Redzbydphi=-L*(Re*really+Im*imagy)/(PI*tempfloat);
Imdzbydphi=-L*(Re*imagy-Im*really)/(PI*tempfloat);

realmov=realerror*Redzbydphi-imagerror*Imdzbydphi;
imagmov=realerror*Imdzbydphi+imagerror*Redzbydphi;

tempfloat=sqrt(realmov*realmov+imagmov*imagmov);
```

```
if(tempfloat>maxmov)
{
realmov=realmov*maxmov/tempfloat;
imagmov=imagmov*maxmov/tempfloat;
}
//printf("%d      %f      %f      %f      %f      %f      %f      %f
%f\n",cycle,realerror,imagerror,theta,psi,complex[0],complex[1],x,y);
x+=realmov;
y+=imagmov;
if((realerror*realerror+imagerror*imagerror)>1.0e-12&&cycle<=100)goto
again;

complex[0]=x;
complex[1]=y;

oldcomplex[0]=x;
oldcomplex[1]=y;

complex[2]=sqrt(Redzbydphi*Redzbydphi+Imdzbydphi*Imdzbydphi);
//}
}

void cosm1exp(float complex[])
{
float Re,Im;
int control;

control=0;
if(complex[0]>=0.0){control=1;complex[0]=-complex[0];}

Re=exp(complex[1])*cos(-complex[0]);
Im=exp(complex[1])*sin(-complex[0]);

real=Re*Re-Im*Im-1.0;
imag=2.0*Re*Im;
squareroot();
real+=Re;
imag+=Im;

recttopolar();
complex[0]=angle;
complex[1]=-log(rad);
if(control==1)complex[1]=-complex[1];
}

void squareroot(void)
{
recttopolar();
angle=angle/2.0;
rad=sqrt(rad);
real=rad*cos(angle);
imag=rad*sin(angle);
}

void recttopolar(void)
{
float PI;
PI=4.0*atan(1.0);

rad=sqrt(real*real+imag*imag);
if(rad>tol)
{
if(real<imag)
{
```

```
angle=acos(real/rad);
if(imag<0.0)angle=2.0*PI-angle;
}
else
{
angle=asin(imag/rad);
if(real<0.0)angle=PI-angle;
if(angle<0.0)angle+=2.0*PI;
}
}
else angle=0.0;
}
```

B2.1 Footbridge Study computer program: Reads B2.0 and produces dxf output files of section and elevation maps

```
#include <math.h>
#include <stdio.h>
#define    surfnum    20000
#define    WireSpacing    3
#define    downsh        40
#define    rightsh        2

extern void SetUpDrawingArea();
//extern void DrawSurface();
extern void FinishOffDrawing();

extern void NewBranching();
extern void expthingy();

void wirescreen(void);
void SetUpDrawingArea(void);
void DrawALine(void);

float theta,psi,psifactor,scale,crot1,srot1,crot2,srot2,
      x,y,z,xplot[surfnum],yplot[surfnum],zplot[surfnum],
      a,b,geometry[2],values[2],complex[3],oldcomplex[2],
      alpha,L,thingy,thetacrit,xline[2],yline[2],zline[2];
int   i,j,m,n,quarter,mapnumber,mfactor,reflection,
      halfW,halfH,intxscreen,intyscreen;

WindowPtr   myWindow;
Rect        theRect;
RGBColor    Colour;

FILE *Wire;

void main(void)
{
mfactor=2*WireSpacing;

psifactor=pi/(1.0*n);

a=2.0/1.5;
b=0.8/1.5;
thetacrit=2.0*sqrt(a*b)/(a+b);
thetacrit=0.5*log((1.0+thetacrit)/(1.0-thetacrit));
L=100.0;
alpha=0.8;

for(mapnumber=1;mapnumber<=2;mapnumber+=1)
{
SetUpDrawingArea();
```

```
if(mapnumber==1)Wire=fopen("Map1.dxf","w");
else Wire=fopen("Map2.dxf","w");
fprintf(Wire,"0\n");
fprintf(Wire,"SECTION\n");
fprintf(Wire,"2\n");
fprintf(Wire,"ENTITIES\n");

for(reflection=-1;reflection<=1;reflection+=2)
{
if(mapnumber!=1||reflection!=-1)
{
for(quarter=1;quarter<=4;quarter+=1)
{
if(quarter==1)
{
if(mapnumber==1){m=4*mfactor;n=10*mfactor;}
if(mapnumber==2){m=12*mfactor;n=10*mfactor;}
//if(mapnumber==2){m=3*mfactor;n=10*mfactor;}
}
if(mapnumber==1&&quarter==3)m=m*(1.0+alpha)/(1.0-alpha);
values[0]=alpha;
values[1]=L;
for(j=0;j<=n;j+=1)
{
for(i=0;i<=m;i+=1)
{
if(mapnumber==1)
{
if(i==0)
{
oldcomplex[0]=L/2.0;
oldcomplex[1]=L/2.0;
if(quarter>=3)oldcomplex[1]=-oldcomplex[1];
}
if(quarter<=2)
{
complex[0]=((1.0-alpha)*pi/2.0)*(1.0-(1.0*i)/(1.0*m));
complex[1]=((1.0-alpha)*pi/2.0)*(1.0*j)/(1.0*m)-1.0;
}
else
{
complex[0]=-((1.0+alpha)*pi/2.0)*(1.0-(1.0*i)/(1.0*m));
complex[1]=((1.0+alpha)*pi/2.0)*(1.0*j)/(1.0*m)-1.0;
if(alpha==0&&complex[0]==0.0)complex[0]=-1.0e-12;
}
expthingy(values,complex,oldcomplex);
x=(1.0-2.0*complex[0]/L);y=2.0*complex[1]/L;z=0.0;
}
if(mapnumber==2)
{
complex[0]=(thetacrit*i)/(10.0*mfactor);
//complex[0]=1.0*thetacrit+(pi/4.0)*(2.0*i-1.0*m)/(1.0*n);
thingy=1.0e-6;
if(quarter==1||quarter==2)
complex[1]=thingy+((pi/2.0)-2.0*thingy)*(1.0*j)/(1.0*n);
else
complex[1]=thingy+(pi/2.0)+1.0*((pi/2.0)-2.0*thingy)*(1.0*j)/(1.0*n);
geometry[0]=a;
geometry[1]=b;
NewBranching(geometry,complex);
x=complex[0];y=reflection*(complex[1]+(a+b)/2.0);z=0.0;
if(i==0)y=0.0;
}

if(quarter==2||quarter==4)x=-x;
```

```
scale=50.0e3;
xplot[(n+1)*i+j]=scale*x;
yplot[(n+1)*i+j]=scale*y;
zplot[(n+1)*i+j]=scale*z;
}
}

wirescreen();
}
}
}
while (!Button());
CloseWindow(myWindow);
fprintf(Wire,"0\n");
fprintf(Wire,"ENDSEC\n");
fprintf(Wire,"0\n");
fprintf(Wire,"EOF\n");
fclose(Wire);
}
}

void wirescreen(void)
{
zline[0]=0.0;zline[1]=0.0;
for(i=0;i<=m;i+=WireSpacing)
{
for(j=0;j<=n-1;j+=1)
{
xline[0]=xplot[(n+1)*(i+0)+(j+0)];
yline[0]=yplot[(n+1)*(i+0)+(j+0)];
xline[1]=xplot[(n+1)*(i+0)+(j+1)];
yline[1]=yplot[(n+1)*(i+0)+(j+1)];
DrawALine();
}
}

for(j=0;j<=n;j+=WireSpacing)
{
for(i=0;i<=m-1;i+=1)
{
xline[0]=xplot[(n+1)*(i+0)+(j+0)];
yline[0]=yplot[(n+1)*(i+0)+(j+0)];
xline[1]=xplot[(n+1)*(i+1)+(j+0)];
yline[1]=yplot[(n+1)*(i+1)+(j+0)];
DrawALine();
}
}
}

void SetUpDrawingArea(void)
{
   InitGraf(&qd.thePort);
   InitFonts();
   InitWindows();
   InitMenus();
   TEInit();
   InitDialogs(0L);
   FlushEvents(everyEvent,0);
   InitCursor();

   halfW=317;halfH=218;

   SetRect(&theRect,rightsh,downsh,rightsh+2*halfW,downsh+2*halfH);
```

```
    myWindow=NewCWindow(0L,&theRect,
    "\pCentre for Lightweight Structures, University of Bath, U.K.",
    true,documentProc,(WindowPtr)-1L,true,0L);
    SetPort(myWindow);
}

void DrawALine(void)
{
intxscreen=halfW+xline[0]/1000.0;
intyscreen=halfH-yline[0]/1000.0;
MoveTo(intxscreen,intyscreen);
intxscreen=halfW+xline[1]/1000.0;
intyscreen=halfH-yline[1]/1000.0;
LineTo(intxscreen,intyscreen);

fprintf(Wire,"0\nLINE\n8\n0\n");
fprintf(Wire,"10\n%f\n",xline[0]);
fprintf(Wire,"20\n%f\n",yline[0]);
fprintf(Wire,"30\n%f\n",zline[0]);
fprintf(Wire,"11\n%f\n",xline[1]);
fprintf(Wire,"21\n%f\n",yline[1]);
fprintf(Wire,"31\n%f\n",zline[1]);
}
```

B2.2 Footbridge Study computer program: Reads B2.0 and B2.1 to produce final dxf files of bridge

```
#include <math.h>
#include <stdio.h>
#define    surfnum    20000
#define     basicfineness   16//6 or 8?? max, 1 for first render with
Joe
                            //2 for final Joe render, 6 for wire bridge
#define    WireSpacing    3
#define    rotation1     -90.0//-62.6
#define    rotation2      90.0
#define    downsh        40
#define    rightsh        2


extern void NewBranching();
extern void expthingy();

extern void dxfSetUp();
extern void dxf3DSurface();
extern void dxfLine();
extern void dxfFinishOff();

void SetUpDrawingArea(void);

void dxfmakesurface(void);
void dxfwireframe(void);

float theta,psi,scale,
      x,y,z,xplot[surfnum],yplot[surfnum],zplot[surfnum],
      a,b,geometry[2],values[2],complex[3],oldcomplex[2],alpha,L,
      val,thingy,thetacrit,bed,wall,value,oldvalue,oldx,oldy,oldz,
      upordownstream,bank,
      span,PI;

int   i,j,m,n,toporbot,
      halfW,halfH,intxscreen,intyscreen;
```

```
WindowPtr   myWindow;
Rect        theRect;
RGBColor    Colour;

void main(void)
{
PI=4.0*atan(1.0);

scale=500.0;
alpha=0.8;
L=250.0;

SetUpDrawingArea();

dxfSetUp();

n=1*basicfineness;
m=6*basicfineness;//Should be even

values[0]=alpha;
values[1]=L;

thingy=1.0e-6;

/*for(span=-1;span<=1;span+=2)
{
for(upordownstream=-1;upordownstream<=1;upordownstream+=2)
{
for(bank=-1;bank<=1;bank+=2)
{*/
for(toporbot=-1;toporbot<=1;toporbot+=2)
{
for(j=0;j<=n;j+=1)
{
//newvalueornot=1;
for(i=0;i<=m;i+=1)
{
//if(newvalueornot==1)
//{
val=1.0-(1.0*i)/(1.0*m);
//val=val*val;

//a=3.0*(1.0+0.4*cos(PI*val));
a=3.0*(1.0+0.4*cos(PI*val));
b=1.5*(1.0+0.4*cos(PI*val));

thetacrit=2.0*sqrt(a*b)/(a+b);
thetacrit=0.5*log((1.0+thetacrit)/(1.0-thetacrit));

//complex[0]=thetacrit-0.3+0.5*cos(PI*val);
//complex[0]=thetacrit+0.3*cos(PI*val);
complex[0]=thetacrit+0.3*cos(PI*val);
if(toporbot==1)
complex[1]=thingy+((PI/2.0)-2.0*thingy)*(1.0-cos((PI*j)/(1.0*n)))/2.0;
else
complex[1]=thingy+(PI/2.0)+((PI/2.0)-2.0*thingy)*(1.0-
cos((PI*j)/(1.0*n)))/2.0;
complex[0]-=0.1*cos(complex[1]);
complex[1]=complex[1]/2.0+(PI/4.0)*sin(complex[1])-
0.4*sin(2.0*complex[1]);

geometry[0]=a;
geometry[1]=b;

NewBranching(geometry,complex);
```

```
x=complex[0];
y=complex[1];

z=(1.0-alpha)*PI*val/2.0;

complex[0]=z;
complex[1]=0.02*y-0.35;

if(i==0)
{
oldcomplex[0]=L/2.0;
oldcomplex[1]=L/2.0;
}
expthingy(values,complex,oldcomplex);

y=complex[0];
//y=(y-L/2.0)*bank;

x=0.015*complex[2]*x;
x=(1.0-0.15*cos(2.0*PI*y/L))*x;

z=complex[1]+(L/PI)*(1.0/2.0)*log((1.0-alpha)/(1.0+alpha));
z=0.3*z;
z-=1.0;

/*value=1.0;

//bed=2.67+3.88-2.0;//2m above MHWS
bed=0.5*(2.67+3.88)+(fabs(y)-L/2.0)/12.0;//Beach
value=z-bed;

if(bank==-1&&toporbot*bank*whichbankfirst==1)
{
wall=L/2.0-asym;
if(value>(wall-y))value=wall-y;
}

if(value<0.0)
{
newvalueornot=0;
x-=(x-oldx)*value/(value-oldvalue);
y-=(y-oldy)*value/(value-oldvalue);
z-=(z-oldz)*value/(value-oldvalue);
}
oldvalue=value;
oldx=x;
oldy=y;
oldz=z;
}
else
{
x=oldx;
y=oldy;
z=oldz;
}*/

xplot[(n+1)*i+j]=1000.0*x;
yplot[(n+1)*i+j]=1000.0*y;
zplot[(n+1)*i+j]=1000.0*z;
//printf("%f   %f     %f\n",x,y,z);

}
}
```

```
Colour.red=65535.0*1.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
RGBForeColor(&Colour);
for(i=0;i<=m;i+=1)
{
for(j=0;j<=n;j+=1)
{
intxscreen=halfW+(yplot[(n+1)*i+j]-1000.0*L/4.0)/scale;
intyscreen=halfH-zplot[(n+1)*i+j]/scale;
if(j==0)MoveTo(intxscreen,intyscreen);
else LineTo(intxscreen,intyscreen);
}
}

Colour.red=65535.0*0.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
RGBForeColor(&Colour);
for(j=0;j<=n;j+=1)
{
for(i=0;i<=m;i+=1)
{
intxscreen=halfW+(yplot[(n+1)*i+j]-1000.0*L/4.0)/scale;
intyscreen=halfH-zplot[(n+1)*i+j]/scale;
if(i==0)MoveTo(intxscreen,intyscreen);
else LineTo(intxscreen,intyscreen);
}
}

for(span=-1.0;span<=1.0;span+=2.0)
{
for(upordownstream=-1.0;upordownstream<=1.0;upordownstream+=2.0)
{
for(bank=-1.0;bank<=1.0;bank+=2.0)
{
dxfmakesurface();
dxfwireframe();
}
}
}
}

while (!Button());

CloseWindow(myWindow);

dxfFinishOff();
}

void dxfmakesurface(void)
{
for(i=0;i<=m-1;i+=1)
{
for(j=0;j<=n-1;j+=1)
{
dxf3DSurface(
upordownstream*xplot[(n+1)*(i+0)+(j+0)],span*1000.0*L/2.0+bank*yplot[(
n+1)*(i+0)+(j+0)],zplot[(n+1)*(i+0)+(j+0)],
upordownstream*xplot[(n+1)*(i+0)+(j+1)],span*1000.0*L/2.0+bank*yplot[(
n+1)*(i+0)+(j+1)],zplot[(n+1)*(i+0)+(j+1)],
upordownstream*xplot[(n+1)*(i+1)+(j+1)],span*1000.0*L/2.0+bank*yplot[(
n+1)*(i+1)+(j+1)],zplot[(n+1)*(i+1)+(j+1)],
upordownstream*xplot[(n+1)*(i+1)+(j+0)],span*1000.0*L/2.0+bank*yplot[(
n+1)*(i+1)+(j+0)],zplot[(n+1)*(i+1)+(j+0)]);
}
}
```

```
}

void dxfwireframe(void)
{
for(i=0;i<=m;i+=WireSpacing)
{
for(j=0;j<=n-1;j+=1)
{
dxfLine(
upordownstream*xplot[(n+1)*(i+0)+(j+0)],
span*1000.0*L/2.0+bank*yplot[(n+1)*(i+0)+(j+0)],
zplot[(n+1)*(i+0)+(j+0)],
upordownstream*xplot[(n+1)*(i+0)+(j+1)],
span*1000.0*L/2.0+bank*yplot[(n+1)*(i+0)+(j+1)],
zplot[(n+1)*(i+0)+(j+1)]);
}
}
for(j=0;j<=n;j+=WireSpacing)
{
for(i=0;i<=m-1;i+=1)
{
dxfLine(
upordownstream*xplot[(n+1)*(i+0)+(j+0)],
span*1000.0*L/2.0+bank*yplot[(n+1)*(i+0)+(j+0)],
zplot[(n+1)*(i+0)+(j+0)],
upordownstream*xplot[(n+1)*(i+1)+(j+0)],
span*1000.0*L/2.0+bank*yplot[(n+1)*(i+1)+(j+0)],
zplot[(n+1)*(i+1)+(j+0)]);
}
}
}

void SetUpDrawingArea(void)
{
   InitGraf(&qd.thePort);
   InitFonts();
   InitWindows();
   InitMenus();
   TEInit();
   InitDialogs(0L);
   FlushEvents(everyEvent,0);
   InitCursor();

   halfW=317;halfH=218;

   SetRect(&theRect,rightsh,downsh,rightsh+2*halfW,downsh+2*halfH);

   myWindow=NewCWindow(0L,&theRect,
   "\pCentre for Lightweight Structures, University of Bath, U.K.",
   true,documentProc,(WindowPtr)-1L,true,0L);
   SetPort(myWindow);
}
```

B2.3 Some Analytical Results

B2.3.1 Analytical method to arrive at an expression for z and $\coth\eta$ from 4.1.1.4(1)

$$2\eta = \log(z + ia) + \log(z + ib) - \log(z - ia) - \log(z - ia)$$

$$2\eta = 2\phi + 2i\psi = \log\frac{(z+ia)(z+ib)}{(z-ia)(z-ib)} = \log\frac{z^2+i(a+b)z-ab}{z^2-i(a+b)z-ab}$$

$$e^{2\eta} = \frac{z^2+i(a+b)z-ab}{z^2-i(a+b)z-ab}$$

$$\left(z^2-i(a+b)z-ab\right)e^{\eta} = \left(z^2+i(a+b)z-ab\right)e^{-\eta}$$

$$\frac{\left(e^{\eta}-e^{-\eta}\right)}{2}z^2 - i(a+b)\frac{\left(e^{\eta}+e^{-\eta}\right)}{2}z - ab\frac{\left(e^{\eta}-e^{-\eta}\right)}{2} = 0$$

$$\sinh\eta z^2 - i(a+b)\cosh\eta z - ab\sinh\eta = 0$$

$$z = \frac{-i(a+b)\cosh\eta \pm \sqrt{-(a+b)^2\cosh^2\eta + 4ab\sinh^2\eta}}{2\sinh\eta}$$

$$= \frac{-i(a+b)\coth\eta \pm \sqrt{-(a+b)^2\coth^2\eta + 4ab}}{2}$$

$$= -i\frac{(a+b)}{2}\left(\coth\eta \pm \sqrt{\coth^2\eta - \frac{4ab}{(a+b)^2}}\right)$$

$$\cosh\eta = \cosh(\phi+i\psi) = \cosh\phi\cos\psi + i\sinh\phi\sin\psi$$

$$\sinh\eta = \sinh(\phi+i\psi) = \sinh\phi\cos\psi + i\cosh\phi\sin\psi$$

$$\coth\eta = \frac{\cosh\eta}{\sinh\eta} = \frac{\cosh\phi\cos\psi + i\sinh\phi\sin\psi}{\sinh\phi\cos\psi + i\cosh\phi\sin\psi}$$

$$= \frac{(\cosh\phi\cos\psi + i\sinh\phi\sin\psi)(\sinh\phi\cos\psi - i\cosh\phi\sin\psi)}{\sinh^2\phi\cos^2\psi + \cosh^2\phi\sin^2\psi}$$

$$= \frac{\cosh\phi\sinh\phi(\cos^2\psi+\sin^2\psi) - i\cos\phi\sin\phi(\cosh^2\psi+\sinh^2\psi)}{\sinh^2\phi\cos^2\psi + \cosh^2\phi\sin^2\psi}$$

$$= \frac{\cosh\phi\sinh\phi - i\cos\phi\sin\phi}{\sinh^2\phi\cos^2\psi + \cosh^2\phi\sin^2\psi}$$

B2.3.2 Analytical method to arrive at the relationship for a uniform stream given in 4.1.1.4(5)

$$i\log\left(\frac{e^{(\alpha+1)\pi\frac{iz}{L}}+e^{(\alpha-1)\pi\frac{iz}{L}}}{2}\right)=i\log\left(e^{\alpha\pi\frac{iz}{L}}\right)+i\log\left(\frac{e^{\pi\frac{iz}{L}}+e^{-\pi\frac{iz}{L}}}{2}\right)$$

$$=i\alpha\pi\frac{iz}{L}+i\log\left(\frac{\left(\cos\frac{\pi z}{L}+i\sin\frac{\pi z}{L}\right)+\left(\cos\frac{\pi z}{L}-i\sin\frac{\pi z}{L}\right)}{2}\right)$$

$$=-\alpha\frac{\pi z}{L}+i\log\left(\cos\frac{\pi z}{L}\right)$$

When y is large and positive, this reduces to $i\alpha\pi\frac{iz}{L}-i\pi\frac{iz}{L}-i\log 2=(1-\alpha)\frac{\pi z}{L}-i\log 2$

When y is large and negative, this reduces to

$$i\alpha\pi\frac{iz}{L}+i\pi\frac{iz}{L}-i\log 2=-(1+\alpha)\frac{\pi z}{L}-i\log 2$$

B3.0 Computer program for wall study

```
#include <math.h>
#include <stdio.h>

#define    surfnum        50
#define    linenum       100
#define    paperscale      1
#define    downsh         40
#define    rightsh         2

void SetUpDrawingArea(void);
void StartPicture(void);
void FinishOffDrawing(void);
void DrawSurface(void);

void dxfSetUp(void);
void dxfmakesurface(void);
void dxfFinishOff();
void potential(void);
void differential(void);
void complexcos(void);
void complexsin(void);
void findwhere(void);
void findangle(void);
```

```
double scale,
      x[surfnum][surfnum],y[surfnum][surfnum],z[surfnum][surfnum],

xplot[surfnum][surfnum],yplot[surfnum][surfnum],zplot[surfnum][surfnum
],
      scalemult,xline[2],yline[2],PI,a,

xanswer,yanswer,xstart[linenum],ystart[linenum],tempcoord[linenum],
      xstarting,ystarting,xworking,yworking,
      rad,real,imag,tempfloat,tempreal,tempimag,coshthingy[21],
      realcos,imagcos,realsin,imagsin,
      phi,psi,realdpotbydz,imagdpotbydz,
      actualphi,actualpsi,errorphi,errorpsi,sine,cosine,angle,

reallogpot,imaglogpot,psilimit,anglefactor,dybydthing,dxbydthing,
      xnorm,ynorm,znorm,red,green,blue,
      bigR,upshift,distancefromtop,
      radius,r0,theta,shapefactor1,shapefactor2,
      psiend,phiend,psifactor;
int   i,j,k,m,n,halfm,
      bayno,across,up,mirrorbay,mirroracross,mirrorup,
      cycle,
      row,nrows,totup,backorfront,

dxfornot,xscreen[4],yscreen[4],side,halfW,halfH,line,tempx,tempy,
      minbayno,maxbayno;

WindowPtr   myWindow;
Rect        theRect;
RGBColor    FillColour,FrameColour,LineColour,Colour;
PolyHandle  Poly;

FILE *Surface;

void main(void)
{
PI=4.0*atan(1.0);
dxfornot=1;
nrows=10;
totup=2*2+1;//Must be odd
minbayno=-7;//Must be odd
maxbayno=8;//Must be even

SetUpDrawingArea();
if(dxfornot==1)dxfSetUp();

scalemult=1.0;
a=100.0;
r0=((maxbayno-minbayno+1.0)*a)/(4.0*PI);
halfm=8;
n=4;
scale=scalemult*paperscale;

m=2*halfm;

anglefactor=2.0*PI*(2.0/(a*(totup+1)));

for(row=1;row<=nrows;row+=1)
{
coshthingy[2*row]=cosh(2.0*PI*row)+1.0;
coshthingy[2*row-1]=cosh(2.0*PI*(2.0*row-1.0)/2.0)+1.0;
}

xworking=a/4.0;
yworking=a/4.0;
```

```
complexcos();
complexsin();
potential();
psiend=actualpsi;
phiend=actualphi;

psifactor=0.5*PI/(4.0*(1.0*halfm-1.0));

xstarting=a/4.0;
ystarting=a/4.0;
for(j=0;j<=n;j+=1)
{
if(j!=0)
{
xstarting=xstart[j-1];
ystarting=ystart[j-1];
}
for(i=halfm;i>=0;i-=1)
{
if(i==0)phi=0.0;
else
{
if(i==halfm)phi=phiend;
else phi=phiend*(1.0*i-0.5)/(1.0*halfm-1.0);
}
if(j==0)psi=0.0;
else psi=psifactor*(1.0*j-0.5);

findwhere();

tempfloat=psi/(psifactor*(1.0*n-0.5));
if(fabs(tempfloat)>1.0)tempfloat=1.0;
z[i][j]=-0.05*a*sqrt(1.0-tempfloat*tempfloat);
x[i][j]=xanswer;
y[i][j]=yanswer;
if(i==halfm)
{
xstart[j]=xanswer;
ystart[j]=yanswer;
}
xstarting=xanswer;
ystarting=yanswer;
if(i==0)y[i][j]=0.0;
if(i==halfm)
{
tempfloat=(x[i][j]-y[i][j])/2.0;
x[i][j]-=tempfloat;
y[i][j]+=tempfloat;
}
if(j==0)
{
tempfloat=(a/2.0-x[i][j]-y[i][j])/2.0;
x[i][j]+=tempfloat;
y[i][j]+=tempfloat;
}
}
}
for(i=halfm-1;i>=0;i-=1)
{
x[m-i][j]=y[i][j];
y[m-i][j]=x[i][j];
z[m-i][j]=z[i][j];
}
}
}

for(backorfront=-1;backorfront<=1;backorfront+=2)
```

```
{
mirrorbay=-1;
for(bayno=2*minbayno;bayno<=2*maxbayno;bayno+=2)
{
mirrorbay=-mirrorbay;
mirroracross=1;
for(across=-1;across<=1;across+=2)
{
mirroracross=-mirroracross;
mirrorup=-1;
for(up=-totup;up<=totup;up+=2)
{
mirrorup=-mirrorup;
for(j=0;j<=n;j+=1)
{
for(i=0;i<=m;i+=1)
{
zplot[i][j]=(x[i][j]-a/4.0)*mirrorup*mirrorbay+a*up/4.0;
xplot[i][j]=(y[i][j]-a/8.0)*mirroracross+a*(1.0*across+2.0*bayno)/8.0;
distancefromtop=(totup*a/2.0-zplot[i][j])/(totup*a);
yplot[i][j]=backorfront*z[i][j]*0.5*(1.0+0.3*exp(5.0*distancefromtop))
;

theta=4.0*PI*xplot[i][j]/((maxbayno-minbayno+1.0)*a);
shapefactor1=0.2;
shapefactor2=0.25;
dxbydthing=cos(theta);
dybydthing=(-shapefactor1*sin(theta)-shapefactor2*2.0*sin(2.0*theta));
tempfloat=r0*(shapefactor1*cos(theta)+shapefactor2*cos(2.0*theta))
+yplot[i][j]*dxbydthing;
xplot[i][j]=r0*sin(theta)-yplot[i][j]*dybydthing;
yplot[i][j]=tempfloat;
}
}
}
if(dxfornot==1)dxfmakesurface();

for(j=0;j<=n;j+=1)
{
for(i=0;i<=m;i+=1)
{
xplot[i][j]=scale*xplot[i][j];
tempfloat=scale*zplot[i][j];
zplot[i][j]=scale*yplot[i][j];
yplot[i][j]=tempfloat;
}
}

DrawSurface();
}
}
}
}

while (!Button());
if(dxfornot==1)dxfFinishOff();
}

void dxfSetUp(void)
{
Surface=fopen("a=bWall.dxf","w");
fprintf(Surface,"0\n");
fprintf(Surface,"SECTION\n");
fprintf(Surface,"2\n");
fprintf(Surface,"ENTITIES\n");
}
```

```
void dxfmakesurface(void)
{
for(i=0;i<=m-1;i+=1)
{
for(j=0;j<=n-1;j+=1)
{
fprintf(Surface,"0\n3DFACE\n8\n0\n");
fprintf(Surface,"10\n%f\n",xplot[i][j]);
fprintf(Surface,"20\n%f\n",yplot[i][j]);
fprintf(Surface,"30\n%f\n",zplot[i][j]);
fprintf(Surface,"11\n%f\n",xplot[i][j+1]);
fprintf(Surface,"21\n%f\n",yplot[i][j+1]);
fprintf(Surface,"31\n%f\n",zplot[i][j+1]);
fprintf(Surface,"12\n%f\n",xplot[i+1][j+1]);
fprintf(Surface,"22\n%f\n",yplot[i+1][j+1]);
fprintf(Surface,"32\n%f\n",zplot[i+1][j+1]);
fprintf(Surface,"13\n%f\n",xplot[i+1][j]);
fprintf(Surface,"23\n%f\n",yplot[i+1][j]);
fprintf(Surface,"33\n%f\n",zplot[i+1][j]);
}
}
}

void dxfFinishOff(void)
{
fprintf(Surface,"0\n");
fprintf(Surface,"ENDSEC\n");
fprintf(Surface,"0\n");
fprintf(Surface,"EOF\n");
fclose(Surface);
}

void complexcos()
{
realcos=+cos(2.0*PI*xworking/a)*cosh(2.0*PI*yworking/a);
imagcos=-sin(2.0*PI*xworking/a)*sinh(2.0*PI*yworking/a);
}

void complexsin()
{
realsin=+sin(2.0*PI*xworking/a)*cosh(2.0*PI*yworking/a);
imagsin=+cos(2.0*PI*xworking/a)*sinh(2.0*PI*yworking/a);
}

void potential()
{
reallogpot=sin(PI*xworking/a)*cosh(PI*yworking/a);
imaglogpot=cos(PI*xworking/a)*sinh(PI*yworking/a);
for(row=1;row<=nrows;row+=1)
{
tempreal=reallogpot*(1.0-(1.0+realcos)/coshthingy[2*row])
        -imaglogpot*(-imagcos/coshthingy[2*row]);
tempimag=reallogpot*(-imagcos/coshthingy[2*row])
        +imaglogpot*(1.0-(1.0+realcos)/coshthingy[2*row]);
reallogpot=tempreal;
imaglogpot=tempimag;

tempreal=+reallogpot*(1.0+(1.0+realcos)/coshthingy[2*row-1])
         +imaglogpot*(imagcos/coshthingy[2*row-1]);
tempimag=-reallogpot*(imagcos/coshthingy[2*row-1])
         +imaglogpot*(1.0+(1.0+realcos)/coshthingy[2*row-1]);
tempfloat=(1.0+(1.0+realcos)/coshthingy[2*row-
1])*(1.0+(1.0+realcos)/coshthingy[2*row-1])
+(imagcos/coshthingy[2*row-1])*(imagcos/coshthingy[2*row-1]);
```

```
reallogpot=tempreal/tempfloat;
imaglogpot=tempimag/tempfloat;
}
tempfloat=sqrt(reallogpot*reallogpot+imaglogpot*imaglogpot);
sine=imaglogpot/tempfloat;
cosine=reallogpot/tempfloat;
findangle();
actualphi=angle;
actualpsi=-log(tempfloat);
}

void findangle()
{
if(fabs(sine)>fabs(cosine))
{
angle=asin(sine);
if(cosine<0.0)angle=PI-angle;
}
else
{
angle=acos(cosine);
if(sine<0.0)angle=-angle;
}
}

void differential()
{
tempfloat=(1.0-realcos)*(1.0-realcos)+imagcos*imagcos;
tempreal=0.5*(1.0-realcos)/tempfloat;
tempimag=0.5*imagcos/tempfloat;
for(row=1;row<=nrows;row+=1)
{
tempfloat=(coshthingy[2*row]-realcos)*(coshthingy[2*row]-
realcos)+imagcos*imagcos;
tempreal+=(coshthingy[2*row]-realcos)/tempfloat;
tempimag+=imagcos/tempfloat;

tempfloat=(coshthingy[2*row-1]+realcos)*(coshthingy[2*row-
1]+realcos)+imagcos*imagcos;
tempreal-=(coshthingy[2*row-1]+realcos)/tempfloat;
tempimag+=imagcos/tempfloat;
}
realdpotbydz=+(2.0*PI/a)*(realsin*tempimag+imagsin*tempreal);
imagdpotbydz=-(2.0*PI/a)*(realsin*tempreal-imagsin*tempimag);
}

void findwhere()
{
if(phi!=0.0||psi!=0.0)
{
xworking=xstarting;
yworking=ystarting;
cycle=0;
again:
complexcos();
complexsin();
potential();
differential();
errorphi=phi-actualphi;
errorpsi=psi-actualpsi;
tempfloat=realdpotbydz*realdpotbydz+imagdpotbydz*imagdpotbydz;
xworking+=(errorphi*realdpotbydz+errorpsi*imagdpotbydz)/tempfloat;
yworking+=(errorpsi*realdpotbydz-errorphi*imagdpotbydz)/tempfloat;
cycle+=1;
```

```
if(cycle<500&&(errorphi*errorphi+errorpsi*errorpsi)>1.0e-18)goto
again;
}
else
{
xworking=a/2.0;
yworking=0.0;
}
xanswer=xworking;
yanswer=yworking;
}

void SetUpDrawingArea()
{
   InitGraf(&qd.thePort);
   InitFonts();
   InitWindows();
   InitMenus();
   TEInit();
   InitDialogs(0L);
   FlushEvents(everyEvent,0);
   InitCursor();

   halfW=317*paperscale;halfH=218*paperscale;

   SetRect(&theRect,rightsh,downsh,rightsh+2*halfW,downsh+2*halfH);

   myWindow=NewCWindow(0L,&theRect,
   "\pCentre for Lightweight Structures, University of Bath, U.K.",
   true,documentProc,(WindowPtr)-1L,true,0L);
   SetPort(myWindow);
}
void DrawSurface(void)
{
for(j=0;j<=n-1;j+=1)
{
for(i=m-1;i>=0;i-=1)
   {
   xscreen[0]=xplot[i][j];yscreen[0]=yplot[i][j];
   xscreen[1]=xplot[i+1][j];yscreen[1]=yplot[i+1][j];
   xscreen[2]=xplot[i+1][j+1];yscreen[2]=yplot[i+1][j+1];
   xscreen[3]=xplot[i][j+1];yscreen[3]=yplot[i][j+1];

   xnorm=(yplot[i+1][j+1]-yplot[i][j])*(zplot[i+1][j]-zplot[i][j+1])
        -(zplot[i+1][j+1]-zplot[i][j])*(yplot[i+1][j]-yplot[i][j+1]);

   ynorm=(zplot[i+1][j+1]-zplot[i][j])*(xplot[i+1][j]-xplot[i][j+1])
        -(xplot[i+1][j+1]-xplot[i][j])*(zplot[i+1][j]-zplot[i][j+1]);

   znorm=(xplot[i+1][j+1]-xplot[i][j])*(yplot[i+1][j]-yplot[i][j+1])
        -(yplot[i+1][j+1]-yplot[i][j])*(xplot[i+1][j]-xplot[i][j+1]);

   if(xnorm!=0.0||ynorm!=0.0||znorm!=0.0)
   {
   znorm=znorm/sqrt(xnorm*xnorm+ynorm*ynorm+znorm*znorm);
   red=fabs(znorm);
   green=fabs(znorm);
   blue=fabs(znorm);

   Poly=OpenPoly();
   MoveTo(halfW+xscreen[0],halfH-yscreen[0]);
   for(side=1;side<=3;side+=1)LineTo(halfW+xscreen[side],halfH-
yscreen[side]);
   LineTo(halfW+xscreen[0],halfH-yscreen[0]);
   ClosePoly();
```

```
   Colour.red=65535.0*red;
   Colour.green=65535.0*green;
   Colour.blue=65535.0*blue;
   RGBForeColor(&Colour);
   PaintPoly(Poly);
   Colour.red=65535.0*1.0;
   Colour.green=65535.0*0.0;
   Colour.blue=65535.0*0.0;
   RGBForeColor(&Colour);
   FramePoly(Poly);

   KillPoly(Poly);
   }
}
}
}
```

B4.0 Computer program for spiral sculpture

```
#include <math.h>
#include <stdio.h>

#define    surfnum      100
#define    linenum      100
#define    downsh        40
#define    rightsh        2

void SetUpDrawingArea(void);
void StartPicture(void);
void FinishOffDrawing(void);
void DrawSurface(void);

void dxfSetUp(void);
void dxfmakesurface(void);
void dxfFinishOff();
void potential(void);
void differential(void);
void complexcos(void);
void complexsin(void);
void findwhere(void);
void findangle(void);

double screenscale,

x[2][surfnum][surfnum],y[2][surfnum][surfnum],z[2][surfnum][surfnum],

xplot[surfnum][surfnum],yplot[surfnum][surfnum],zplot[surfnum][surfnum
],
      xline[2],yline[2],PI,a,c,

xanswer,yanswer,xstart[linenum],ystart[linenum],tempcoord[linenum],
      xstarting,ystarting,xworking,yworking,
      rad,real,imag,tempfloat,tempreal,tempimag,coshthingy[21],
      realcos,imagcos,realsin,imagsin,
      phi,psi,realdpotbydz,imagdpotbydz,psiangle,
      actualphi,actualpsi,errorphi,errorpsi,sine,cosine,angle,
      reallogpot,imaglogpot,
      phiend,psiend,
      xnorm,ynorm,znorm,red,green,blue,
      bigR,upshift,
      theta,eta,anglefactor,
      floatxtemp,floatytemp,floatztemp,
      etamax,Radius,BendAngle,Twist;
```

```
int   i,j,k,halfm,m,n,
      numberofbays,bayno,mirroracross,up,mirrorup,bayswitch,
      mirrorupstart,mirrorupstop,
      cycle,
      row,nrows,totup,backorfront,whichone,

dxfornot,xscreen[4],yscreen[4],side,halfW,halfH,line,tempx,tempy;

WindowPtr   myWindow;
Rect        theRect;
RGBColor    FillColour,FrameColour,LineColour,Colour;
PolyHandle  Poly;

FILE *Surface;

void main(void)
{
PI=4.0*atan(1.0);
dxfornot=1;
nrows=10;
totup=1;
numberofbays=4;//Must be divisible by 4
SetUpDrawingArea();

screenscale=0.02;
a=100.0;
c=10.0e3;
halfm=20;
m=2*halfm;
n=6;

anglefactor=2.0*PI*(2.0/(a*(totup+1)));

for(row=1;row<=nrows;row+=1)
{
coshthingy[2*row]=cosh(2.0*PI*row)+1.0;
coshthingy[2*row-1]=cosh(2.0*PI*(2.0*row-1.0)/2.0)+1.0;
}

for(whichone=0;whichone<=1;whichone+=1)
{
xworking=a/4.0;
yworking=a/4.0;
complexcos();
complexsin();
potential();
psiend=actualpsi;
phiend=actualphi;

if(dxfornot==1)dxfSetUp();
xstarting=a/4.0;
ystarting=a/4.0;
for(j=0;j<=n;j+=1)
{
if(j!=0)
{
xstarting=xstart[j-1];
ystarting=ystart[j-1];
}
for(i=halfm;i>=0;i-=1)
{
phi=phiend*(1.0-cos((PI*i)/(2.0*halfm)));
psiangle=(PI*j)/(2.0*n);
psiangle=(PI/2.0)*(1.0-cos(psiangle));
if(whichone==0)psi=psiend*(phi/phiend)+0.1*sin(psiangle);
```

```
else psi=psiend*(phi/phiend)+0.1*sin(psiangle);

findwhere();

z[whichone][i][j]=0.2*cos(psiangle);
x[whichone][i][j]=xanswer;
y[whichone][i][j]=yanswer;
if(i==halfm)
{
xstart[j]=xanswer;
ystart[j]=yanswer;
}
xstarting=xanswer;
ystarting=yanswer;
if(i==0)y[whichone][i][j]=0.0;
if(i==halfm)
{
tempfloat=(x[whichone][i][j]-y[whichone][i][j])/2.0;
x[whichone][i][j]-=tempfloat;
y[whichone][i][j]+=tempfloat;
}
if(j==0)
{
tempfloat=(a/2.0-x[whichone][i][j]-y[whichone][i][j])/2.0;
x[whichone][i][j]+=tempfloat;
y[whichone][i][j]+=tempfloat;
}
}
for(i=halfm-1;i>=0;i-=1)
{
x[whichone][m-i][j]=y[whichone][i][j];
y[whichone][m-i][j]=x[whichone][i][j];
z[whichone][m-i][j]=z[whichone][i][j];
}
}
}
}

bayswitch=1;
for(bayno=1;bayno<=numberofbays;bayno+=1)
{
if(bayswitch==1)bayswitch=0;
else bayswitch=1;
for(up=1;up<=totup+bayswitch;up+=1)
{
if(up==totup+bayswitch&&bayswitch==1)whichone=1;
else whichone=0;
mirrorupstart=-1;mirrorupstop=1;
if(bayswitch==1)
{
if(up==1)mirrorupstart=1;
if(up==totup+bayswitch)mirrorupstop=-1;
}

for(mirrorup=mirrorupstart;mirrorup<=mirrorupstop;mirrorup+=2)
{
for(mirroracross=-1;mirroracross<=1;mirroracross+=2)
{
for(backorfront=-1;backorfront<=1;backorfront+=2)
{
for(j=0;j<=n;j+=1)
{
for(i=0;i<=m;i+=1)
{
xplot[i][j]=x[whichone][i][j]*mirrorup+a*up-(bayswitch*a)/2.0-a/2.0;
yplot[i][j]=y[whichone][i][j]*mirroracross+a*(4.0*bayno)/8.0;
```

```
etamax=1.0*4.0*PI*(totup*a/2.0+a/4.0)/(numberofbays*a);
eta=etamax*xplot[i][j]/(totup*a);
theta=4.0*PI*(yplot[i][j]-a/2.0)/(numberofbays*a);

floatxtemp=sin(theta)*cosh(eta);
floatytemp=cos(theta)*sinh(eta);
floatztemp=0.1*backorfront*z[whichone][i][j]*cosh(eta);

Radius=sqrt(floatxtemp*floatxtemp+floatytemp*floatytemp);
BendAngle=asin(0.9*Radius/
sqrt(sin(theta)*sin(theta)*cosh(etamax)*cosh(etamax)
    +cos(theta)*cos(theta)*sinh(etamax)*sinh(etamax)));
zplot[i][j]=(10.0+floatztemp)*c*cos(BendAngle);
Twist=zplot[i][j]/(6.0*c);
Twist=Twist*Twist;
xplot[i][j]=(1.0+floatztemp)*c*sin(BendAngle)*
            (floatxtemp*cos(Twist)-floatytemp*sin(Twist))/Radius;
yplot[i][j]=(1.0+floatztemp)*c*sin(BendAngle)*
            (floatxtemp*sin(Twist)+floatytemp*cos(Twist))/Radius;

//zplot[i][j]-=52000.0;
zplot[i][j]-=60000.0;
zplot[i][j]=2.0*zplot[i][j];

//xplot[i][j]=floatxtemp*c;
//yplot[i][j]=floatytemp*c;
//zplot[i][j]=floatztemp*c;
}
}
if(dxfornot==1)dxfmakesurface();

for(j=0;j<=n;j+=1)
{
for(i=0;i<=m;i+=1)
{
xplot[i][j]=screenscale*xplot[i][j];
tempfloat=screenscale*zplot[i][j];
zplot[i][j]=screenscale*yplot[i][j];
yplot[i][j]=tempfloat;
yplot[i][j]=zplot[i][j];
}
}
DrawSurface();
}
}
}
}
}
}
if(dxfornot==1)dxfFinishOff();

while (!Button());
}

void dxfSetUp(void)
{
Surface=fopen("Sculpture.dxf","w");
fprintf(Surface,"0\n");
fprintf(Surface,"SECTION\n");
fprintf(Surface,"2\n");
fprintf(Surface,"ENTITIES\n");
}

void dxfmakesurface(void)
{
```

```
for(i=0;i<=m-1;i+=1)
{
for(j=0;j<=n-1;j+=1)
{
fprintf(Surface,"0\n3DFACE\n8\n0\n");
fprintf(Surface,"10\n%f\n",xplot[i][j]);
fprintf(Surface,"20\n%f\n",yplot[i][j]);
fprintf(Surface,"30\n%f\n",zplot[i][j]);
fprintf(Surface,"11\n%f\n",xplot[i][j+1]);
fprintf(Surface,"21\n%f\n",yplot[i][j+1]);
fprintf(Surface,"31\n%f\n",zplot[i][j+1]);
fprintf(Surface,"12\n%f\n",xplot[i+1][j+1]);
fprintf(Surface,"22\n%f\n",yplot[i+1][j+1]);
fprintf(Surface,"32\n%f\n",zplot[i+1][j+1]);
fprintf(Surface,"13\n%f\n",xplot[i+1][j]);
fprintf(Surface,"23\n%f\n",yplot[i+1][j]);
fprintf(Surface,"33\n%f\n",zplot[i+1][j]);
}
}
}
}

void dxfFinishOff(void)
{
fprintf(Surface,"0\n");
fprintf(Surface,"ENDSEC\n");
fprintf(Surface,"0\n");
fprintf(Surface,"EOF\n");
fclose(Surface);
}

void complexcos()
{
realcos=+cos(2.0*PI*xworking/a)*cosh(2.0*PI*yworking/a);
imagcos=-sin(2.0*PI*xworking/a)*sinh(2.0*PI*yworking/a);
}

void complexsin()
{
realsin=+sin(2.0*PI*xworking/a)*cosh(2.0*PI*yworking/a);
imagsin=+cos(2.0*PI*xworking/a)*sinh(2.0*PI*yworking/a);
}

void potential()
{
reallogpot=sin(PI*xworking/a)*cosh(PI*yworking/a);
imaglogpot=cos(PI*xworking/a)*sinh(PI*yworking/a);
for(row=1;row<=nrows;row+=1)
{
tempreal=reallogpot*(1.0-(1.0+realcos)/coshthingy[2*row])
        -imaglogpot*(-imagcos/coshthingy[2*row]);
tempimag=reallogpot*(-imagcos/coshthingy[2*row])
        +imaglogpot*(1.0-(1.0+realcos)/coshthingy[2*row]);
reallogpot=tempreal;
imaglogpot=tempimag;

tempreal=+reallogpot*(1.0+(1.0+realcos)/coshthingy[2*row-1])
         +imaglogpot*(imagcos/coshthingy[2*row-1]);
tempimag=-reallogpot*(imagcos/coshthingy[2*row-1])
         +imaglogpot*(1.0+(1.0+realcos)/coshthingy[2*row-1]);
tempfloat=(1.0+(1.0+realcos)/coshthingy[2*row-
1])*(1.0+(1.0+realcos)/coshthingy[2*row-1])
+(imagcos/coshthingy[2*row-1])*(imagcos/coshthingy[2*row-1]);
reallogpot=tempreal/tempfloat;
imaglogpot=tempimag/tempfloat;
}
```

```
tempfloat=sqrt(reallogpot*reallogpot+imaglogpot*imaglogpot);
sine=imaglogpot/tempfloat;
cosine=reallogpot/tempfloat;
findangle();
actualphi=angle;
actualpsi=-log(tempfloat);
}

void findangle()
{
if(fabs(sine)>fabs(cosine))
{
angle=asin(sine);
if(cosine<0.0)angle=PI-angle;
}
else
{
angle=acos(cosine);
if(sine<0.0)angle=-angle;
}
}

void differential()
{
tempfloat=(1.0-realcos)*(1.0-realcos)+imagcos*imagcos;
tempreal=0.5*(1.0-realcos)/tempfloat;
tempimag=0.5*imagcos/tempfloat;
for(row=1;row<=nrows;row+=1)
{
tempfloat=(coshthingy[2*row]-realcos)*(coshthingy[2*row]-
realcos)+imagcos*imagcos;
tempreal+=(coshthingy[2*row]-realcos)/tempfloat;
tempimag+=imagcos/tempfloat;

tempfloat=(coshthingy[2*row-1]+realcos)*(coshthingy[2*row-
1]+realcos)+imagcos*imagcos;
tempreal-=(coshthingy[2*row-1]+realcos)/tempfloat;
tempimag+=imagcos/tempfloat;
}
realdpotbydz=+(2.0*PI/a)*(realsin*tempimag+imagsin*tempreal);
imagdpotbydz=-(2.0*PI/a)*(realsin*tempreal-imagsin*tempimag);
}

void findwhere()
{
if(phi!=0.0||psi!=0.0)
{
xworking=xstarting;
yworking=ystarting;
cycle=0;
again:
complexcos();
complexsin();
potential();
differential();
errorphi=phi-actualphi;
errorpsi=psi-actualpsi;
tempfloat=realdpotbydz*realdpotbydz+imagdpotbydz*imagdpotbydz;
xworking+=(errorphi*realdpotbydz+errorpsi*imagdpotbydz)/tempfloat;
yworking+=(errorpsi*realdpotbydz-errorphi*imagdpotbydz)/tempfloat;
cycle+=1;
if(cycle<100&&(errorphi*errorphi+errorpsi*errorpsi)>1.0e-30)goto
again;
}
else
```

```
{
xworking=a/2.0;
yworking=0.0;
}
xanswer=xworking;
yanswer=yworking;
}

void SetUpDrawingArea()
{
   InitGraf(&qd.thePort);
   InitFonts();
   InitWindows();
   InitMenus();
   TEInit();
   InitDialogs(0L);
   FlushEvents(everyEvent,0);
   InitCursor();

   halfW=317;halfH=218;

   SetRect(&theRect,rightsh,downsh,rightsh+2*halfW,downsh+2*halfH);

   myWindow=NewCWindow(0L,&theRect,
   "\pCentre for Lightweight Structures, University of Bath, U.K.",
   true,documentProc,(WindowPtr)-1L,true,0L);
   SetPort(myWindow);
}

void DrawSurface(void)
{
for(j=0;j<=n-1;j+=1)
{
for(i=m-1;i>=0;i-=1)
   {
   xscreen[0]=xplot[i][j];yscreen[0]=yplot[i][j];
   xscreen[1]=xplot[i+1][j];yscreen[1]=yplot[i+1][j];
   xscreen[2]=xplot[i+1][j+1];yscreen[2]=yplot[i+1][j+1];
   xscreen[3]=xplot[i][j+1];yscreen[3]=yplot[i][j+1];

   xnorm=(yplot[i+1][j+1]-yplot[i][j])*(zplot[i+1][j]-zplot[i][j+1])
        -(zplot[i+1][j+1]-zplot[i][j])*(yplot[i+1][j]-yplot[i][j+1]);

   ynorm=(zplot[i+1][j+1]-zplot[i][j])*(xplot[i+1][j]-xplot[i][j+1])
        -(xplot[i+1][j+1]-xplot[i][j])*(zplot[i+1][j]-zplot[i][j+1]);

   znorm=(xplot[i+1][j+1]-xplot[i][j])*(yplot[i+1][j]-yplot[i][j+1])
        -(yplot[i+1][j+1]-yplot[i][j])*(xplot[i+1][j]-xplot[i][j+1]);

   if(xnorm!=0.0||ynorm!=0.0||znorm!=0.0)
   {
   znorm=znorm/sqrt(xnorm*xnorm+ynorm*ynorm+znorm*znorm);
   red=fabs(znorm);
   green=fabs(znorm);
   blue=fabs(znorm);

   Poly=OpenPoly();
   MoveTo(halfW+xscreen[0],halfH-yscreen[0]);
   for(side=1;side<=3;side+=1)LineTo(halfW+xscreen[side],halfH-
yscreen[side]);
   LineTo(halfW+xscreen[0],halfH-yscreen[0]);
   ClosePoly();

   Colour.red=65535.0*red;
   Colour.green=65535.0*green;
```

```
    Colour.blue=65535.0*blue;
    RGBForeColor(&Colour);
    PaintPoly(Poly);
    Colour.red=65535.0*1.0;
    Colour.green=65535.0*0.0;
    Colour.blue=65535.0*0.0;
    RGBForeColor(&Colour);
    FramePoly(Poly);

    KillPoly(Poly);
    }
}
}
}
```

Appendix C1.0 Rest Zone Computer Programs

C1.1 Outline C++ Syntax relating to the Program for the Schematic Torus

Details of the structure and syntax for computer programming in C++ and Visual C++ for Windows based applications are discussed in various manuals such as that by Schildt (1994) and Gurewich (1996) and the reader is referred to these texts for a complete and detailed tutorial in C++ programming.

C++ programming is presented here only in relation to the application in hand to communicate an overall understanding of its use in this particular context.

The description given here of the structure and syntax of a typical computer program has been colour-coded, in order to group together certain types of instructions and to distinguish between code instructions and explanatory notes.

Terms appearing in blue in the following program are standard C++ code and usually involve instructions that the computer recognises and implements. For example the statement **#include** asks the compiler to refer to certain libraries or header files which help it to understand various terms in the program. One such file is the **<math.h>** header file, which enables the compiler to carry out instructions involving mathematical expressions by drawing on the vocabulary of the math.h library function. The **<iostream.h>** file always appears because it relates to a pre-prepared support file that allows the C++ compiler to use input and display output information. The **<fstream.h>** is a header file for both input and output.

The code **#define** instructs the compiler to substitute a named identifier (or user-defined term) with a character sequence which will remain constant throughout the program. In this case the named identifiers are **mPlus1** and **nPlus1**, which will be substituted with the character sequence **201** each time they are encountered in the program or source file. This means that changes to the value of a character sequence need only be made once at the start of the program where the identifier is defined, for that change to take effect wherever the named identifier appears in the rest of the program.

Where **int** appears, a variable of type integer is declared and appropriate space required to store it is allocated.

The code **float** or **double** asks the compiler to assign larger memory locations or storage boxes to keep computations involving floating point numerals or numbers requiring double precision, and declares that these data types will be used in the function. Values for these integers, floats and doubles (which could either be variables or constants) are then assigned before or after the start of a function.

If a variable is declared outside function blocks, it is a **global** variable and all functions have access to its value. If it is declared within a function it is a **local** variable and operations on it are only performed by that function.

The prefix **ofstream** asks for output data from a function to be written to the file enclosed in brackets, in this case to Rest("Torus.dxf").

The code **main(void)** is the name reserved for the function which begins the program execution. A program must always contain a main function. In some instances the prefix **int** or **void** before a function name declares its return type. For example where **int main(void)** appears, it implies that the main function returns values of integer type, although if nothing is specified, an integer type return value is assumed by default. In the case of **void main(void)** the main function does not return a value. The parentheses after the function name distinguish it from variables and specify whether or not the function has any parameters. **(void)** after the function name indicates that the function has no parameters. When the parentheses contain variables, the function will have arguments or values passed to it, which are known as the parameters of that function.

A program can contain more than one function, but the name or prototype of that function must be declared globally, before it is called or defined. One function can call another to operate but the *definition* of that other function cannot itself be embedded within the function calling it, only the call to it.

Apart from references to standard header files, named identifiers and the reference marking the point at which a function is defined, all C++ statements, including calls to functions must end with a **semi-colon**.

The code **for**, initialises a loop, setting the first value of a given variable to 0 or as otherwise specified, and instructs the compiler to execute successive cycles of a computation or function whilst increasing the value assigned to each variable in specified increments until the function conditions are no longer met. So **for(j=0;j<=n;j+=1)**, the loop will start with a value of j = 0 and continue to implement the function for all values of j less than or equal to n in increments of 1.**+ =1** denotes that the magnitude of the increment is 1.

The code **return 0;** terminates a function and the zero following it indicates that the function is ending normally.

Finally, terms in green are printed or output as part of the dxf file when it is created. If the statement is enclosed in speech marks, it is printed to the output file.

The above summarises the essential features of a simple C++ program. Other brief comments will be given in the appropriate section of the program as they become necessary.

C1.2 Computer Program to generate the Schematic Torus

The notation // is used at the start of a single-line comment, the notation /* is used at the start of a multi-line comment and */ closes a multi-line comment. Comments denoted in this way are ignored by the compiler. If a multi-line comment is likely to interfere with a program instruction because it appears on the same line as the instruction, the single-line comment marker will be used for each line forming a part of that comment to prevent the compiler from reading it with the program instruction.

//This is the start of the program; comments in red are not part of the program.

```
#include <math.h>                //header file for mathematical
                                 computations
#include <fstream.h>             //header file for input and output
#include <iostream.h>            //support file for input and output
#define  mPlus1     201          //named identifier and character
                                 sequence
#define  nPlus1     201
double   x[mPlus1][nPlus1],      //memory blocks reserving space for 201
         y[mPlus1][nPlus1],      //by 201 elements are allocated for x,
         z[mPlus1][nPlus1],      /*y and z which are declared here to be
                                 of data type double. Square brackets
                                 enclose and specify the array extents of
                                 x, y and z co-ordinates whose values
                                 extend from 0 to 200 as specified by the
                                 character sequence above. The array
                                 extents are also known as the index
                                 values. Index values indicate the
                                 position or address of an element in an
                                 array*/

              PI,theta,phi,R;    //PI, theta, phi and R are declared here

int           i,j,m,n,step;      /*memory blocks are allocated here for
                                 i, j, m, n and step which are user-
                                 defined terms declared here to be
                                 variables of integer type*/

ofstream Rest("Torus.dxf");      /*A file called Torus.dxf is opened to
                                 input data generated by the Rest
                                 function which returns its values to
                                 this file*/

int main(void)                   /*main function is defined here commencing
```

```
                                with the opening curly bracket*/
{
PI=4.0*atan(1.0);               //The value of PI is declared here
R=12000.0;                      //The value of R is declared here

r=R/2.0;                        //The value of r is declared here
step=10;                        //The value of step is declared here

m=10*step;                      //The value of m is declared here

/*The next statement ensures that the function cycle is aborted when
the value of m exceeds the value allowed by the storage space
allocated to it. This is achieved by the use of the if(  ) statement
with a condition specified in brackets. cout<< is the code requesting
console or screen output of the statement contained between the speech
marks notifying the user that m has exceeded its limits. \n is the
code for a new line and is not printed to the screen. The curly
brackets enclose a logical unit or code block which is associated with
the if statement and return 0 asks the function to end normally*/

if(m>mPlus1-1){cout<<"m too big\n";return 0;}

n=5*step;                       //The value of n is declared here

if(n>nPlus1-1){cout<<"n too big\n";return 0;}

/*The next part of the program calculates x, y and z co-ordinates
using the trigonometric relationships described in 4.2.1. Here, i and
j are used to initialise the arrays of function cycles by setting
their values from 0 to 200 in increments of 1. In the next set of
instructions it is important to note that phi and theta are expressed
in terms of i,j,m and n so that phi and theta vary in increments
corresponding to the increments of i,j, m and n*/
```

```
for(j=0;j<=n;j+=1)  //loading the arrays of j initialises the cycle
   {
   phi=(2.0*PI*j)/(1.0*n);     /*phi is varied in equal increments of j
                               over the total array cycle of n*/

for(i=0;i<=m;i+=1)
   {
   theta=(2.0*PI*i)/(1.0*m);  /*theta is is varied in equal increments
                               of i over the total array cycle of m*/

   x[i][j]=(R+r*cos(phi))*cos(theta);      //cos and sin are understood
                                           //by the math.h
   y[i][j]=r*sin(phi);         //header file
   z[i][j]=(R+r*cos(phi))*sin(theta);
   }                           //the main function is terminated here
}                              // with the final closing curly bracket
/*The next part of the program writes the arrays of the x, y and z co-
ordinates to the output file. 0, SECTION, 2, ENTITIES is the standard
format for dxf output indicating the type of data and layer on which
that data appears*/
Rest<<"0\rSECTION\r2\rENTITIES\r";  // \r is the code for a carriage
                                    // return

/*The next set of instructions writes the co-ordinates of the rings to
the dxf file using the i array as it travels along rows of j. In this
case i is incremented by step, which means that only the data of every
tenth co-ordinate of i is sent to the dxf file resulting in a
schematic series of rings.*/

   for(i=0;i<=m-step;i+=step) //i initialises the loop for the i array
{                        //of co-ordinates and is incremented by step
```

```
   for(j=0;j<=n-1;j+=1) //rather than 1 which is used to control the
                        //coarseness of data drawn to the dxf file
{
      Rest<<"0\rLINE\r8\rRings\r";
      Rest<<"10\r"<<x[i][j]<<"\r";
      Rest<<"20\r"<<y[i][j]<<"\r";
      Rest<<"30\r"<<z[i][j]<<"\r";
      Rest<<"11\r"<<x[i][j+1]<<"\r";
                        //incrementing the j array moves the cycle
      Rest<<"21\r"<<y[i][j+1]<<"\r";
                        //on to successive rows of j once each full
      Rest<<"31\r"<<z[i][j+1]<<"\r";
                        //loop of i is complete up until j maximum
}

/*The next set of instructions write the co-ordinates of the hoops to
the dxf file using the j array as it travels along rows of i. In this
case j is incremented by step so that only the data of every tenth co-
ordinate of j is sent to the dxf file creating a schematic series of
hoops*/

 for(j=0;j<=n-step;j+=step)
{
   for(i=0;i<=m-1;i+=1)

     {
      Rest<<"0\rLINE\r8\rHoops\r";
      Rest<<"10\r"<<x[i][j]<<"\r";
      Rest<<"20\r"<<y[i][j]<<"\r";
      Rest<<"30\r"<<z[i][j]<<"\r";
      Rest<<"11\r"<<x[i+1][j]<<"\r";
                        //incrementing the i array moves the cycle
      Rest<<"21\r"<<y[i+1][j]<<"\r";
```

```
                        //on to successive rows of i once each full
      Rest<<"31\r"<<z[i+1][j]<<"\r";
                        //loop of j is complete up until i maximum
      }
}
/*dxf files are terminated with the 'end of file' syntax in green.
Rest.close(), return 0 and the closing curly brace close the input
file*/

      Rest<<"0\rENDSEC\r0\rEOF\r";Rest.close();

return 0;

}
```

C1.3 Rest Zone detailed Computer Program

```
#include <math.h>
#include <fstream.h>
#include <iostream.h>

#define  mPlus1      121
#define  nPlus1       31
#define  MaxNodes  20000
#define  halfW       317
#define  halfH       218
#define  downsh       40
#define  rightsh       2

double
x[4][2][mPlus1][nPlus1],y[4][2][mPlus1][nPlus1],z[4][2][mPlus1][nPlus1
],
         by[4][2][mPlus1][nPlus1],bz[4][2][mPlus1][nPlus1],
         theta[4][2][mPlus1][nPlus1],phi[4][2][mPlus1][nPlus1],
         Nodex[2][MaxNodes],Nodey[2][MaxNodes],Nodez[2][MaxNodes],
         PI,tempdouble,ActualLength[2],NominalLength[2],
         xav,yav,zav,xtolerance,
         ribdepthover2,ribwidthover2,
         scale,xshift,yshift,xplot[2],yplot[2],
         xGround,yGround,zGround,
         R,r[2],
         q,Q,f,fdash,
         deltay1,deltaz1,deltay2,deltaz2,
         ellipse[2],ellipsey,ellipsefactor,
         groundcontrol,myfloor,
         step,lambda,stepriser,stepHt,
         area[2],areax,areay,areaz,
         vector1x,vector1y,vector1z,vector2x,vector2y,vector2z,
         xcontrol;

int      i,j,m[2],n,
```

```
        i1,i2,i3,i4,j1,j2,j3,j4,
        intxscreen[2],intyscreen[2],
        toporbottom,backorfront,flip,flop,Newton,
        dxfornot,deformornot,Part,Sector,
        NumNodes[2],
        CurrentNode,NodeNumber[4][2][mPlus1][nPlus1],
        Previousj,whichi,whichj,TrimOrNot,otheri,
        PanelNumber,BeamNumber,
        RibInters,between,VariableSpacing,NewtonOrNot;

WindowPtr   myWindow;
Rect        theRect;
RGBColor    Colour;

ofstream Ribs("CladRib.dxf");
ofstream Sect("Sections.dxf");
ofstream Staad("Rest.std");
ofstream Areas("Areas of Cladding");

void findcoords(void);
void findtangent(void);
void innerdeform(void);
void makeastep(void);
void Ground(void);
void jthing(void);
void NumberInc(void);
void dxfClad(void);
void dxfRibs(void);
void dxfSections(void);
void OpenMacWindow(void);
void MacLine(void);
void CloseMacWindow(void);
double xfunction(double functionangle);

int main(void)
{
PI=4.0*atan(1.0);

//cout<<"Do you want variable rib spacing? Yes = 1.\n";
//cin>>VariableSpacing;
VariableSpacing=1;

//cout<<"Do you want to deform the torus? Yes = 1.\n";
//cin>>deformornot;
deformornot=1;

cout<<"Do you want dxf files and data files? Yes = 1.\n";
cin>>dxfornot;

for(;;)
{
cout<<"How many sections between ribs (must be zero or one)?\n";
cin>>between;
if(between==0||between==1)break;
}

between+=1;

ActualLength[0]=31000.0;
ActualLength[1]=30000.0;

if(VariableSpacing==1)
{
RibInters=30;
}
```

```
else
{
RibInters=26;
NominalLength[0]=31200.0;
NominalLength[1]=ActualLength[1];
}

m[0]=RibInters*between+16*(between-1)+12*(between-1);

if(m[0]>mPlus1-1){cout<<"m too big at "<<m[0]<<"\n";return 0;}

cout<<"What value of n do you want? Usually it is 30 for drawings and
5 for analysis.\n";
cin>>n;

if(n>nPlus1-1){cout<<"n too big\n";return 0;}

cout<<"Do you want to trim the bits below ground? 1 = yes.\n";
cin>>TrimOrNot;
//TrimOrNot=1;

xtolerance=10.0;

if(VariableSpacing==1)
r[0]=ActualLength[0]*(1.0-
xfunction(PI*(2.0+1.0/6.0)/(1.0*RibInters)))/4.0;
else
r[0]=(ActualLength[0]-
(NominalLength[0]*(2.0+1.0/6.0))/(0.5*RibInters))/4.0;

R=ActualLength[0]/2.0-r[0];
r[1]=ActualLength[1]/2.0-R;

m[1]=m[0];

ellipse[0]=0.6*(R+r[0]);
ellipse[1]=0.6*(R+r[1]);

scale=40.0;
xshift=-80.0;
yshift=150.0;

ribdepthover2=300.0;
ribwidthover2=75.0;

for(Part=0;Part<=1;Part+=1)
{
for(Sector=0;Sector<=3;Sector+=1)
{
  for(i=0;i<=m[Part];i+=1)
   {
      for(j=0;j<=n;j+=1)
      if(TrimOrNot==1)NodeNumber[Sector][Part][i][j]=-1;
                          else NodeNumber[Sector][Part][i][j]=0;
   }
}
}

for(Sector=0;Sector<=3;Sector+=1)
{
if(Sector==0||Sector==3)backorfront=0;else backorfront=1;
if(Sector==0||Sector==2)toporbottom=0;else toporbottom=1;

  Part=0;findcoords();findtangent();
```

```
  Part=1;findcoords();findtangent();if(deformornot==1)innerdeform();
}

for(Part=0;Part<=1;Part+=1)
{
for(i=0;i<=m[Part];i+=1)
{
xav=0.0;
for(Sector=0;Sector<=3;Sector+=1)xav+=x[Sector][Part][i][0]+x[Sector][
Part][i][n];
xav=xav/8.0;
for(Sector=0;Sector<=3;Sector+=1)
{
x[Sector][Part][i][0]=xav;
x[Sector][Part][i][n]=xav;
}

yav=(y[0][Part][i][0]+y[2][Part][i][0])/2.0;y[0][Part][i][0]=yav;y[2][
Part][i][0]=yav;
zav=(z[0][Part][i][0]+z[2][Part][i][0])/2.0;z[0][Part][i][0]=zav;z[2][
Part][i][0]=zav;

yav=(y[1][Part][i][0]+y[3][Part][i][0])/2.0;y[1][Part][i][0]=yav;y[3][
Part][i][0]=yav;
zav=(z[1][Part][i][0]+z[3][Part][i][0])/2.0;z[1][Part][i][0]=zav;z[3][
Part][i][0]=zav;

if(fabs(fabs(xav)-(R-r[Part]))<xtolerance)//The  x  tolerance  is  to
force meeting
{
yav=(y[0][Part][i][n]+y[1][Part][i][n]+y[2][Part][i][n]+y[3][Part][i][
n])/4.0;
y[0][Part][i][n]=yav;y[1][Part][i][n]=yav;y[2][Part][i][n]=yav;y[3][Pa
rt][i][n]=yav;

zav=(z[0][Part][i][n]+z[1][Part][i][n]+z[2][Part][i][n]+z[3][Part][i][
n])/4.0;
z[0][Part][i][n]=zav;z[1][Part][i][n]=zav;z[2][Part][i][n]=zav;z[3][Pa
rt][i][n]=zav;
}
if(fabs(xav)<R-r[Part])
{
yav=(y[0][Part][i][n]+y[2][Part][i][n])/2.0;y[0][Part][i][n]=yav;y[2][
Part][i][n]=yav;
zav=(z[0][Part][i][n]+z[2][Part][i][n])/2.0;z[0][Part][i][n]=zav;z[2][
Part][i][n]=zav;

yav=(y[1][Part][i][n]+y[3][Part][i][n])/2.0;y[1][Part][i][n]=yav;y[3][
Part][i][n]=yav;
zav=(z[1][Part][i][n]+z[3][Part][i][n])/2.0;z[1][Part][i][n]=zav;z[3][
Part][i][n]=zav;
}
if(fabs(xav)>R-r[Part])
{
yav=(y[0][Part][i][n]+y[3][Part][i][n])/2.0;y[0][Part][i][n]=yav;y[3][
Part][i][n]=yav;
zav=(z[0][Part][i][n]+z[3][Part][i][n])/2.0;z[0][Part][i][n]=zav;z[3][
Part][i][n]=zav;

yav=(y[1][Part][i][n]+y[2][Part][i][n])/2.0;y[1][Part][i][n]=yav;y[2][
Part][i][n]=yav;
zav=(z[1][Part][i][n]+z[2][Part][i][n])/2.0;z[1][Part][i][n]=zav;z[2][
Part][i][n]=zav;
}
}
}
```

```
}

for(Part=0;Part<=1;Part+=1)
{
CurrentNode=0;
for(i=0;i<=m[Part];i+=1)
{
  for(Sector=0;Sector<=3;Sector+=1)
   {
      for(j=n;j>=0;j-=1)
      {
      if(j!=n)Previousj=j+1;else Previousj=n;

if(z[Sector][Part][i][Previousj]>0.0&&z[Sector][Part][i][j]<0.0&&TrimO
rNot==1)
      {
      y[Sector][Part][i][j]-=
      z[Sector][Part][i][j]*(y[Sector][Part][i][Previousj]-
y[Sector][Part][i][j])
                              /(z[Sector][Part][i][Previousj]-
z[Sector][Part][i][j]);
      z[Sector][Part][i][j]=0.0;
      }

if((z[Sector][Part][i][Previousj]>=0.0&&z[Sector][Part][i][j]>=0.0)||T
rimOrNot!=1)
      NumberInc();
      else
      {
      if(i==1||i==m[Part]-1)
      {
      NumberInc();
      if(i==1)otheri=0;
      if(i==m[Part]-1)otheri=m[Part];
      x[Sector][Part][i][j]-=
      z[Sector][Part][i][j]*(x[Sector][Part][otheri][j]-
x[Sector][Part][i][j])
                                    /(z[Sector][Part][otheri][j]-
z[Sector][Part][i][j]);}
      z[Sector][Part][i][j]=0.0;
      }
      }
   }
}
NumNodes[Part]=CurrentNode;
cout<<"Part "<<Part<<", number of nodes = "<<NumNodes[Part]<<"\n";
if(CurrentNode>MaxNodes){cout<<"Too many nodes\n";return 0;}

for(Sector=0;Sector<=3;Sector+=1)
{
   for(i=0;i<=m[Part];i+=1)
   {
      for(j=0;j<=n;j+=1)
      {
      Nodex[Part][NodeNumber[Sector][Part][i][j]-
1]=x[Sector][Part][i][j];
      Nodey[Part][NodeNumber[Sector][Part][i][j]-
1]=y[Sector][Part][i][j];
      Nodez[Part][NodeNumber[Sector][Part][i][j]-
1]=z[Sector][Part][i][j];
      }
   }
}
if(dxfornot==1&&Part==0)
{
```

```
Staad<<"STAAD SPACE STRUT\r";
Staad<<"INPUT WIDTH 72\r";
Staad<<"UNIT METER KNS\r";
Staad<<"JOINT COORDINATES\r";
for(CurrentNode=1;CurrentNode<=NumNodes[Part];CurrentNode+=1)
{
Staad<<CurrentNode;
Staad<<"    "<<Nodey[Part][CurrentNode-1]/1000.0;
Staad<<"    "<<Nodez[Part][CurrentNode-1]/1000.0;
Staad<<"    "<<Nodex[Part][CurrentNode-1]/1000.0;
Staad<<"\r";
}
}
}

OpenMacWindow();
if(dxfornot==1)
{
Ribs<<"0\rSECTION\r2\rENTITIES\r";
Sect<<"0\rSECTION\r2\rENTITIES\r";
Ground();
}

for(Sector=0;Sector<=3;Sector+=1)
{
  for(i=0;i<=m[0];i+=1)
   {
      for(j=0;j<=n;j+=1)
      {
      if(i<m[0])
        {
        xplot[0]=x[Sector][0][i][j];xplot[1]=x[Sector][0][i+1][j];
        yplot[0]=z[Sector][0][i][j];yplot[1]=z[Sector][0][i+1][j];

Colour.red=65535.0*1.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
        RGBForeColor(&Colour);
        if(xplot[0]>=0.0&&xplot[1]>=0.0)MacLine();
        }
      if(j<n)
        {
        xplot[0]=x[Sector][0][i][j];xplot[1]=x[Sector][0][i][j+1];
        yplot[0]=z[Sector][0][i][j];yplot[1]=z[Sector][0][i][j+1];

Colour.red=65535.0*1.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
        RGBForeColor(&Colour);
        if(xplot[0]>=0.0&&xplot[1]>=0.0)MacLine();

        xplot[0]=y[Sector][0][i][j];xplot[1]=y[Sector][0][i][j+1];

Colour.red=65535.0*0.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
        RGBForeColor(&Colour);
        MacLine();

        if(i<=m[1])
        {
        xplot[0]=y[Sector][1][i][j];xplot[1]=y[Sector][1][i][j+1];
        yplot[0]=z[Sector][1][i][j];yplot[1]=z[Sector][1][i][j+1];

Colour.red=65535.0*0.0;Colour.green=65535.0*1.0;Colour.blue=65535.0*0.
0;
        RGBForeColor(&Colour);
        MacLine();
```

```
        }
        }
      }
   }
}

if(dxfornot==1)
{
area[0]=0.0;
area[1]=0.0;
dxfSections();
Staad<<"MEMBER INCIDENCES\r";
dxfRibs();
Staad<<"ELEMENT INCIDENCES SHELL\r";
dxfClad();
Areas<<"Area of outer cladding ="<<area[0]<<" square metres\r";
Areas<<"Area of inner cladding including floor = "<<area[1]<<" square
metres\r";
}

if(dxfornot==1)
{
Ribs<<"0\rENDSEC\r0\rEOF\r";Ribs.close();
Sect<<"0\rENDSEC\r0\rEOF\r";Sect.close();
Staad.close();
}
CloseMacWindow();

cout<<"Finished\n";
return 0;
}

void dxfClad(void)
{
for(Part=0;Part<=1;Part+=1)
{
PanelNumber=BeamNumber;
   for(i=0;i<=m[Part]-1;i+=1)
   {
   for(Sector=0;Sector<=3;Sector+=1)
   {
      for(j=0;j<=n-1;j+=1)
      {
      if(NodeNumber[Sector][Part][i][j+1]!=-1
        ||NodeNumber[Sector][Part][i+1][j+1]!=-1)
      {
      whichi=i;whichj=j;    jthing();i1=whichi;j1=whichj;
      whichi=i;whichj=j+1;  jthing();i2=whichi;j2=whichj;
      whichi=i+1;whichj=j+1;jthing();i3=whichi;j3=whichj;
      whichi=i+1;whichj=j;  jthing();i4=whichi;j4=whichj;
      if(z[Sector][Part][i1][j1]!=0.0||
         z[Sector][Part][i2][j2]!=0.0||
         z[Sector][Part][i3][j3]!=0.0||
         z[Sector][Part][i4][j4]!=0.0)
      {
      if(Part==0)Ribs<<"0\r3DFACE\r8\rOuterCladding\r";
            else Ribs<<"0\r3DFACE\r8\rInnerCladding\r";
      Ribs<<"10\r"<<x[Sector][Part][i1][j1]<<"\r";
      Ribs<<"20\r"<<y[Sector][Part][i1][j1]<<"\r";
      Ribs<<"30\r"<<z[Sector][Part][i1][j1]<<"\r";
      Ribs<<"11\r"<<x[Sector][Part][i2][j2]<<"\r";
      Ribs<<"21\r"<<y[Sector][Part][i2][j2]<<"\r";
      Ribs<<"31\r"<<z[Sector][Part][i2][j2]<<"\r";
      Ribs<<"12\r"<<x[Sector][Part][i3][j3]<<"\r";
      Ribs<<"22\r"<<y[Sector][Part][i3][j3]<<"\r";
```

```
      Ribs<<"32\r"<<z[Sector][Part][i3][j3]<<"\r";
      Ribs<<"13\r"<<x[Sector][Part][i4][j4]<<"\r";
      Ribs<<"23\r"<<y[Sector][Part][i4][j4]<<"\r";
      Ribs<<"33\r"<<z[Sector][Part][i4][j4]<<"\r";

      vector1x=(x[Sector][Part][i1][j1]+x[Sector][Part][i2][j2]-
(x[Sector][Part][i3][j3]+x[Sector][Part][i4][j4]))/2.0;
      vector2x=(x[Sector][Part][i2][j2]+x[Sector][Part][i3][j3]-
(x[Sector][Part][i4][j4]+x[Sector][Part][i1][j1]))/2.0;

      vector1y=(y[Sector][Part][i1][j1]+y[Sector][Part][i2][j2]-
(y[Sector][Part][i3][j3]+y[Sector][Part][i4][j4]))/2.0;
      vector2y=(y[Sector][Part][i2][j2]+y[Sector][Part][i3][j3]-
(y[Sector][Part][i4][j4]+y[Sector][Part][i1][j1]))/2.0;

      vector1z=(z[Sector][Part][i1][j1]+z[Sector][Part][i2][j2]-
(z[Sector][Part][i3][j3]+z[Sector][Part][i4][j4]))/2.0;
      vector2z=(z[Sector][Part][i2][j2]+z[Sector][Part][i3][j3]-
(z[Sector][Part][i4][j4]+z[Sector][Part][i1][j1]))/2.0;

      areax=vector1y*vector2z-vector2y*vector1z;
      areay=vector1z*vector2x-vector2z*vector1x;
      areaz=vector1x*vector2y-vector2x*vector1y;

      area[Part]+=sqrt(areax*areax+areay*areay+areaz*areaz)/1.0e6;

      if(Part==0)
      {
      if(NodeNumber[Sector][Part][i1][j1]
       !=NodeNumber[Sector][Part][i2][j2]
      &&NodeNumber[Sector][Part][i2][j2]
       !=NodeNumber[Sector][Part][i3][j3]
       &&NodeNumber[Sector][Part][i3][j3]
       !=NodeNumber[Sector][Part][i1][j1])
      {
      PanelNumber+=1;
      Staad<<PanelNumber;
      Staad<<"    "<<NodeNumber[Sector][Part][i1][j1];
      Staad<<"    "<<NodeNumber[Sector][Part][i2][j2];
      Staad<<"    "<<NodeNumber[Sector][Part][i3][j3];
      Staad<<"\r";
      }

      if(NodeNumber[Sector][Part][i3][j3]
       !=NodeNumber[Sector][Part][i4][j4]
       &&NodeNumber[Sector][Part][i4][j4]
       !=NodeNumber[Sector][Part][i1][j1]
       &&NodeNumber[Sector][Part][i1][j1]
       !=NodeNumber[Sector][Part][i3][j3])
      {
      PanelNumber+=1;
      Staad<<PanelNumber;
      Staad<<"    "<<NodeNumber[Sector][Part][i3][j3];
      Staad<<"    "<<NodeNumber[Sector][Part][i4][j4];
      Staad<<"    "<<NodeNumber[Sector][Part][i1][j1];
      Staad<<"\r";
      }
      }
      }
      }
   }
   }
}
}
```

```
void dxfSections(void)
{
for(Part=0;Part<=1;Part+=1)
{
  for(i=0;i<=m[Part];i+=1)
    {
    for(Sector=0;Sector<=3;Sector+=1)
    {
       for(j=0;j<=n-1;j+=1)
       {
       if(NodeNumber[Sector][Part][i][j]!=-1
        &&NodeNumber[Sector][Part][i][j+1]!=-1)
       {
       if(Part==0)Sect<<"0\rLINE\r8\rOuterCladding\r";
                  else Sect<<"0\rLINE\r8\rInnerCladding\r";

       Sect<<"10\r"<<Nodex[Part][NodeNumber[Sector][Part][i][j]-
1]<<"\r";
       Sect<<"20\r"<<Nodey[Part][NodeNumber[Sector][Part][i][j]-
1]<<"\r";
       Sect<<"30\r"<<Nodez[Part][NodeNumber[Sector][Part][i][j]-
1]<<"\r";
       Sect<<"11\r"<<Nodex[Part][NodeNumber[Sector][Part][i][j+1]-
1]<<"\r";
       Sect<<"21\r"<<Nodey[Part][NodeNumber[Sector][Part][i][j+1]-
1]<<"\r";
       Sect<<"31\r"<<Nodez[Part][NodeNumber[Sector][Part][i][j+1]-
1]<<"\r";

       //The above replaced the following, purely as a check on
numbering

       /*Sect<<"10\r"<<x[Sector][Part][i][j]<<"\r";
       Sect<<"20\r"<<y[Sector][Part][i][j]<<"\r";
       Sect<<"30\r"<<z[Sector][Part][i][j]<<"\r";
       Sect<<"11\r"<<x[Sector][Part][i][j+1]<<"\r";
       Sect<<"21\r"<<y[Sector][Part][i][j+1]<<"\r";
       Sect<<"31\r"<<z[Sector][Part][i][j+1]<<"\r";*/
       }
       }
    }
    }
}
}

void dxfRibs(void)
{
Part=0;
BeamNumber=0;
for(i=between;i<=m[0]-between;i+=between)
{
if(between==1||
   (2.0*i-m[0]<-26&&2.0*i-m[0]>-72)||
   (2.0*i-m[0]>+26&&2.0*i-m[0]<+72)||
   2.0*i-m[0]==0||
   2.0*i-m[0]==-4||
   2.0*i-m[0]==+4||
   2.0*i-m[0]==-16||
   2.0*i-m[0]==+16||
   2.0*i-m[0]==-76||
   2.0*i-m[0]==+76
   )
{
for(Sector=0;Sector<=3;Sector+=1)
```

```
{
for(flop=-1;flop<=1;flop+=2)
{
  for(flip=-1;flip<=1;flip+=2)
  {

      for(j=0;j<=n-1;j+=1)
      {
      if(NodeNumber[Sector][Part][i][j]!=-1
       &&NodeNumber[Sector][Part][i][j+1]!=-1)
      {
      deltay1=+bz[Sector][Part][i][j+0]*ribdepthover2;
      deltaz1=-by[Sector][Part][i][j+0]*ribdepthover2;

      deltay2=+bz[Sector][Part][i][j+1]*ribdepthover2;
      deltaz2=-by[Sector][Part][i][j+1]*ribdepthover2;

      Ribs<<"0\r3DFACE\r8\rRibs\r";

      Ribs<<"10\r"<<x[Sector][Part][i][j]+flip*ribwidthover2<<"\r";
      Ribs<<"20\r"<<y[Sector][Part][i][j]+(-1.0+flip)*deltay1<<"\r";
      Ribs<<"30\r"<<z[Sector][Part][i][j]+(-1.0+flip)*deltaz1<<"\r";
      Ribs<<"11\r"<<x[Sector][Part][i][j+1]+flip*ribwidthover2<<"\r";
      Ribs<<"21\r"<<y[Sector][Part][i][j+1]+(-1.0+flip)*deltay2<<"\r";
      Ribs<<"31\r"<<z[Sector][Part][i][j+1]+(-1.0+flip)*deltaz2<<"\r";


Ribs<<"12\r"<<x[Sector][Part][i][j+1]+flip*flop*ribwidthover2<<"\r";
      Ribs<<"22\r"<<y[Sector][Part][i][j+1]+(-1.0-
flip*flop)*deltay2<<"\r";
      Ribs<<"32\r"<<z[Sector][Part][i][j+1]+(-1.0-
flip*flop)*deltaz2<<"\r";

Ribs<<"13\r"<<x[Sector][Part][i][j]+flip*flop*ribwidthover2<<"\r";
      Ribs<<"23\r"<<y[Sector][Part][i][j]+(-1.0-
flip*flop)*deltay1<<"\r";
      Ribs<<"33\r"<<z[Sector][Part][i][j]+(-1.0-
flip*flop)*deltaz1<<"\r";

      BeamNumber+=1;
      Staad<<BeamNumber;
      Staad<<"    "<<NodeNumber[Sector][Part][i][j];
      Staad<<"    "<<NodeNumber[Sector][Part][i][j+1];
      Staad<<"\r";
      }
      }
}
}
}
}
}

void OpenMacWindow(void)
{
   InitGraf(&qd.thePort);
   InitFonts();
   InitWindows();
   InitMenus();
   TEInit();
   InitDialogs(0L);
   FlushEvents(everyEvent,0);
   InitCursor();

   SetRect(&theRect,rightsh,downsh,rightsh+2*halfW,downsh+2*halfH);
```

```
    myWindow=NewCWindow(0L,&theRect,
    "\pCentre for Lightweight Structures, University of Bath, U.K.",
    true,documentProc,(WindowPtr)-1L,true,0L);
    SetPort(myWindow);
}

void MacLine(void)
{
intxscreen[0]=halfW+xplot[0]/scale+xshift;
intxscreen[1]=halfW+xplot[1]/scale+xshift;

intyscreen[0]=halfH-yplot[0]/scale+yshift;
intyscreen[1]=halfH-yplot[1]/scale+yshift;

MoveTo(intxscreen[0],intyscreen[0]);
LineTo(intxscreen[1],intyscreen[1]);
}

void CloseMacWindow(void)
{
if(dxfornot!=1)while (!Button());
CloseWindow(myWindow);
}

void Ground(void)
{
xGround=2.0*(R+r[0]);
yGround=4.0*(R+r[0]);
zGround=0.0*(R+r[0]);

      Ribs<<"0\r3DFACE\r8\rGround\r";
      Ribs<<"10\r"<<xGround<<"\r";
      Ribs<<"20\r"<<yGround<<"\r";
      Ribs<<"30\r"<<zGround<<"\r";
      Ribs<<"11\r"<<-xGround<<"\r";
      Ribs<<"21\r"<<yGround<<"\r";
      Ribs<<"31\r"<<zGround<<"\r";
      Ribs<<"12\r"<<-xGround<<"\r";
      Ribs<<"22\r"<<-yGround<<"\r";
      Ribs<<"32\r"<<zGround<<"\r";
      Ribs<<"13\r"<<xGround<<"\r";
      Ribs<<"23\r"<<-yGround<<"\r";
      Ribs<<"33\r"<<zGround<<"\r";
}

void findtangent(void)
{
   for(i=1;i<=m[Part]-1;i+=1)
   {
      for(j=0;j<=n;j+=1)
      {
      j1=j-1;if(j1<0)j1=0;
      j2=j+1;if(j2>n)j2=n;

      by[Sector][Part][i][j]=y[Sector][Part][i][j2]-
y[Sector][Part][i][j1];
      bz[Sector][Part][i][j]=z[Sector][Part][i][j2]-
z[Sector][Part][i][j1];


tempdouble=sqrt(by[Sector][Part][i][j]*by[Sector][Part][i][j]+bz[Secto
r][Part][i][j]*
bz[Sector][Part][i][j]);
```

```
      by[Sector][Part][i][j]=by[Sector][Part][i][j]/tempdouble;
      bz[Sector][Part][i][j]=bz[Sector][Part][i][j]/tempdouble;

      if(toporbottom==1)
      {
      by[Sector][Part][i][j]=-by[Sector][Part][i][j];
      bz[Sector][Part][i][j]=-bz[Sector][Part][i][j];
      }

      if(backorfront==1)
      {
      by[Sector][Part][i][j]=-by[Sector][Part][i][j];
      bz[Sector][Part][i][j]=-bz[Sector][Part][i][j];
      }
      }
    }
}

void findcoords(void)
{
   for(i=0;i<=m[Part];i+=1)
   {
   q=1.0;
   xcontrol=1.0*i-0.5*m[Part];
   if(between==2)
   {
   if(xcontrol>2.0)
   {
   if(xcontrol<14.0)xcontrol=2.0+(xcontrol-2.0)/3.0;
   else
   {
   if(xcontrol<35.0)xcontrol=xcontrol-8.0;
   else xcontrol=27.0+(xcontrol-35.0)/3.0;
   }
   }
   if(xcontrol<-2.0)
   {
   if(xcontrol>-14.0)xcontrol=-2.0+(xcontrol+2.0)/3.0;
   else
   {
   if(xcontrol>-35.0)xcontrol=xcontrol+8.0;
   else xcontrol=-27.0+(xcontrol+35.0)/3.0;
   }
   }
   }
   xcontrol=xcontrol/((1.0*RibInters)*(1.0*between));

      for(j=0;j<=n;j+=1)
      {
      if(VariableSpacing==1)

x[Sector][Part][i][j]=(ActualLength[Part]/2.0)*xfunction(PI*xcontrol);
      else
      {
      x[Sector][Part][i][j]=NominalLength[Part]*xcontrol;

if(x[Sector][Part][i][j]>ActualLength[Part]/2.0)x[Sector][Part][i][j]=
ActualLength[Part]/2.0;
      if(x[Sector][Part][i][j]<-
ActualLength[Part]/2.0)x[Sector][Part][i][j]=-ActualLength[Part]/2.0;
      }

      Q=(1.0-cos((PI*j)/(1.0*n)))/2.0;

      NewtonOrNot=1;
```

```
      if(NewtonOrNot==1)
      {
      Q=1.0-(1.0-Q)*(1.0-Q);
      for(Newton=1;Newton<=30;Newton+=1)
      {
      f=(q+1.0)*((R+r[Part]*q)*(R+r[Part]*q)-
x[Sector][Part][i][j]*x[Sector][Part][i][j])-2.0*(1.0-
Q)*((R+r[Part])*(R+r[Part])-
x[Sector][Part][i][j]*x[Sector][Part][i][j]);
      fdash=((R+r[Part]*q)*(R+r[Part]*q)-
x[Sector][Part][i][j]*x[Sector][Part][i][j])+
2.0*(q+1.0)*r[Part]*(R+r[Part]*q);
      q=q-f/fdash;
      }
      }
      else
      {
      if(fabs(x[Sector][Part][i][j])<=R-r[Part])
      q=1.0-2.0*Q;
      else
      q=1.0-2.0*Q*(R+r[Part]-
fabs(x[Sector][Part][i][j]))/(2.0*r[Part]);
      }

      if(q>+1.0)q=+1.0;
      if(q<-1.0)q=-1.0;
      phi[Sector][Part][i][j]=acos(q);
      if(backorfront==1)phi[Sector][Part][i][j]=-
phi[Sector][Part][i][j];
      y[Sector][Part][i][j]=r[Part]*sin(phi[Sector][Part][i][j]);
      z[Sector][Part][i][j]=sqrt(fabs((R+r[Part]*q)*(R+r[Part]*q)-
x[Sector][Part][i][j]*x[Sector][Part][i][j]));
      if(toporbottom==1)z[Sector][Part][i][j]=-z[Sector][Part][i][j];


if(fabs(z[Sector][Part][i][j])<fabs(x[Sector][Part][i][j]))theta[Secto
r][Part][i][j]=
atan(fabs(z[Sector][Part][i][j]/x[Sector][Part][i][j]));
      else theta[Sector][Part][i][j]=
PI/2.0-atan(fabs(x[Sector][Part][i][j]/z[Sector][Part][i][j]));
      if(x[Sector][Part][i][j]<0.0)theta[Sector][Part][i][j]=PI-
theta[Sector][Part][i][j];
      if(z[Sector][Part][i][j]<0.0)theta[Sector][Part][i][j]=-
theta[Sector][Part][i][j];

      if(deformornot==1)
      {

tempdouble=(x[Sector][Part][i][j]*x[Sector][Part][i][j]+z[Sector][Part
][i][j]
*z[Sector][Part][i][j])/((R+r[Part])*(R+r[Part]));
      if(tempdouble<1.0)ellipsey=ellipse[Part]*sqrt(1.0-tempdouble);
      else ellipsey=0.0;
      if(backorfront==1)ellipsey=-ellipsey;

      tempdouble=(1.0-q)/2.0;

if(tempdouble>0.0)ellipsefactor=pow(tempdouble,3.0*(1.2+sin(theta[Sect
or][Part][i][j])));
      else ellipsefactor=0.0;

y[Sector][Part][i][j]=y[Sector][Part][i][j]*ellipsefactor+ellipsey*(1.
0-ellipsefactor);
```

```
      z[Sector][Part][i][j]=z[Sector][Part][i][j]-0.4*(R-
r[0])*tanh(0.5*z[Sector][Part][i][j]/(R-r[0]));

      //z[Sector][Part][i][j]=1.0*(1.0*z[Sector][Part][i][j]-
0.44*z[Sector][Part][i][j]*z[Sector][Part][i][j]/(R+r[0]));
      z[Sector][Part][i][j]=1.0*(1.0*z[Sector][Part][i][j]-
0.4*z[Sector][Part][i][j]*z[Sector][Part][i][j]/(R+r[0]));

      z[Sector][Part][i][j]+=0.03*(R+r[0])*pow((1.0-
q)/2.0,6.0)*(1.0+sin(theta[Sector][Part][i][j]));


y[Sector][Part][i][j]+=0.3*y[Sector][Part][i][j]*z[Sector][Part][i][j]
/(R+r[0])*
(1.0-sin(theta[Sector][Part][i][j]));

      y[Sector][Part][i][j]-=0.2*y[Sector][Part][i][j]*pow((1.0-
sin(theta[Sector][Part][i][j]))/2.0,5.0);

      z[Sector][Part][i][j]-
=0.3*x[Sector][Part][i][j]*x[Sector][Part][i][j]/(R+r[0]);

  //z[Sector][Part][i][j]-=0.2*(R*R*R*R-(x[Sector][Part][i][j]-
R)*(x[Sector][Part][i][j]-R)
      //
*(x[Sector][Part][i][j]+R)*(x[Sector][Part][i][j]+R))/(R*R*R);

      z[Sector][Part][i][j]+=6100.0;

      y[Sector][Part][i][j]=0.97*y[Sector][Part][i][j];

      //y[Sector][Part][i][j]-
=0.1*y[Sector][Part][i][j]*z[Sector][Part][i][j]/(R+r[0]);

      z[Sector][Part][i][j]-
=0.2*x[Sector][Part][i][j]*x[Sector][Part][i][j]/(R+r[0]);


z[Sector][Part][i][j]+=0.2*x[Sector][Part][i][j]*x[Sector][Part][i][j]
*x[Sector][Part][i][j]
      *x[Sector][Part][i][j]/((R+r[0])*(R+r[0])*(R+r[0]));
      }
      }
   }
}

void innerdeform(void)
     {
      for(i=0;i<=m[Part];i+=1)
      {
      for(j=0;j<=n;j+=1)
      {

groundcontrol=2000.0/cosh(3.0*z[Sector][Part][i][j]/(R+r[Part]));

      myfloor=z[Sector][Part][i][j]-600.0
      -
500.0*(1.0+x[Sector][Part][i][j]/8000.0+y[Sector][Part][i][j]/8000.0)

/(1.0+x[Sector][Part][i][j]*x[Sector][Part][i][j]/(3000.0*3000.0));


z[Sector][Part][i][j]+=(sqrt(myfloor*myfloor+groundcontrol*groundcontr
ol)-myfloor)/2.0;
```

```
      step=0.0;

      stepHt=1800.0;stepriser=400.0;lambda=10.0;makeastep();
      stepHt=1400.0;stepriser=400.0;lambda=10.0;makeastep();
      stepHt=1000.0;stepriser=200.0;lambda=5.0;makeastep();

      z[Sector][Part][i][j]+=step;
      }
      }
     }

void makeastep(void)
{
step-=(z[Sector][Part][i][j]-
stepHt)/(1.0+cosh(lambda*(z[Sector][Part][i][j]-
stepHt)/(stepriser/2.0)+1.0)/cosh(lambda));
}

void jthing(void)
{
for(;;)
{
if(NodeNumber[Sector][Part][whichi][whichj]!=-1)break;
else whichj+=1;
}
}

void NumberInc(void)
{
if(i!=0&&i!=m[Part])
{
   if(j==n&&fabs(fabs(x[Sector][Part][i][j])-(R-r[Part]))<xtolerance)
   {
   if(Sector==0)
      {
      CurrentNode+=1;
      NodeNumber[Sector][Part][i][j]=CurrentNode;
      }
   else NodeNumber[Sector][Part][i][j]=NodeNumber[0][Part][i][j];
   }
   else
   {
      if(Sector==0||Sector==1||(j!=0&&j!=n))
      {
      CurrentNode+=1;
      NodeNumber[Sector][Part][i][j]=CurrentNode;
      }
      else
      {

if(Sector==2&&j==0)NodeNumber[Sector][Part][i][j]=NodeNumber[0][Part][
i][j];

if(Sector==3&&j==0)NodeNumber[Sector][Part][i][j]=NodeNumber[1][Part][
i][j];
      if(j==n)
      {
      if(fabs(x[Sector][Part][i][j])<R-r[Part])
      {

if(Sector==2)NodeNumber[Sector][Part][i][j]=NodeNumber[0][Part][i][j];

if(Sector==3)NodeNumber[Sector][Part][i][j]=NodeNumber[1][Part][i][j];
      }
      if(fabs(x[Sector][Part][i][j])>R-r[Part])
```

```
      {

if(Sector==2)NodeNumber[Sector][Part][i][j]=NodeNumber[1][Part][i][j];

if(Sector==3)NodeNumber[Sector][Part][i][j]=NodeNumber[0][Part][i][j];
      }
      }
      }
   }
}
else
{
     if(Sector==0&&j==n)
      {
      CurrentNode+=1;
      NodeNumber[Sector][Part][i][j]=CurrentNode;
      }
      else NodeNumber[Sector][Part][i][j]=NodeNumber[0][Part][i][n];
}
}

double xfunction(double functionangle)
{
return sin(functionangle)-(1.0/8.0)*sin(2.0*functionangle);
//This gives third derivative zero at functionangle = 0
}
```

Appendix D1.0 British Museum Roof

D1.1 British Museum Roof

Introduction

Following a design competition, Foster and Partners were appointed in the spring of 1997 to design the roof over the Museum courtyard. The proposed design was a triangulated steel grid-shell structure laid out in a configuration of inter-locking spirals. The roof was to be glazed with either single or double-glazing to be determined by a consideration of the internal environment.

Dr Chris Williams at the University of Bath was appointed to generate information regarding the geometry of the roof. This research contract was associated with other work being carried out by Buro Happold and Williams involving the non-linear structural analysis of the roof structure as a whole.

The geometrical solution sought to resolve the disparity between the circular geometry of the Reading Room at the centre of the Museum Court and the rectilinear geometry around its perimeter. A spiral grid was introduced as a geometric device to resolve this problem. A view of the final roof structure is shown in figure D1a.

Figure D1.1a Bird's eye view of the steel grid shell for the British Museum Great Court Roof

Geometrical Constraints

The British Museum Great Court is a rectangular space with an overall width of 73m and length of 97m. The Great Court houses the Sydney Smirke Reading Room, which is a circular drum with an overall diameter of 45m. The Reading Room has a dome roof supported on cast iron vaulted ribs. The internal finish of the Reading Room ceiling is papier maché with gilded edging, which has been fully restored.

The Great Court roof has symmetry about its north-south axis but the Reading room is off set relative to the centre of the Court by 2.6m to its north end, resulting in asymmetry along the east-west axis. For the purposes of shape-finding and geometrical analysis, the centre of the Reading room was taken to be the origin of the setting out geometry.

Sliding bearings are used around the perimeter of the Court to prevent horizontal loads from being transferred onto the existing structure. A clearance of 150mm, which takes account of deflection, was required over the highest pediment at the North Portico, which in turn determined the level of the roof perimeter. No existing elements around the perimeter were to be altered. In addition, the node positions around the Court perimeter were fixed, as were the positions of the nodes at the corners. Given these constraints, a lattice was required to span the residual space between the circle and perimeter court.

Foster and Partners approached Williams, firstly because the precise geometry of the curved lattice was difficult to describe using conventional CAD techniques, and secondly because the discrepancies arising at perimeter conditions where panels finished arbitrarily, proved difficult to resolve using manual drawing techniques.

Structural Considerations

Satisfactory grid-shell geometry relies on an adequate ratio between the overall span of the lattice relative to its rise. Ideally, a grid-shell should not be too flat and outward thrusts developing at boundary conditions need to be contained and diverted back into the structure. An added constraint was that the highest point on the dome had to fall within St. Paul's heights and had also to comply with certain other planning restrictions, in order to prevent the new roof from being visible behind the existing museum when viewed from the street.

Horizontal ties would have counteracted the high horizontal thrusts and would also have minimised the need for larger section sizes, but Fosters rejected this option.

Outward thrusts developing in each sector of the roof were transferred to members located at the perimeter of the roof, whose behaviour was similar to that of a horizontal truss. The forces from the trusses in each quadrant emerged as tensions along the

perimeter members of adjacent sectors. This effect was repeated continuously around the whole roof. These principles are outlined in figure D1.1b.

The members at the centre ring and perimeter rectangle, being the principal boundary members, had the largest section sizes. In order to maintain crispness of the steel sections, and in order also to keep member sizes in general to a minimum, Foster and Partners were keen for the hollow rectangular members to be fabricated from steel plate rather than from rolled hollow section, which had certain other advantages.

Non-standard steel plate of variable thickness made it easier to accommodate the requirements for different section sizes, thus helping to keep the weight of members down. The thickness of standard rolled hollow section would have been more difficult to control, and would not have permitted the tapering of members where a transition occurred from large to small sections without additional machining and welding.

The stiffness, strength and consequently the weight of members, is critical in resisting buckling and deformation, however, the design and strength of the node connections had to match that of the members to avoid any vulnerability occurring at the nodes. Nodes were fully welded and had to accommodate the fact that the members do not lie in one plane. The steel and glass roof contractors, Waagner-Biró, achieved this by flame-cutting the nodes using an adapted welding robot according to mathematically generated geometrical data.

D2.0 Defining the Roof Geometry

D2.1 Roof Starter Grid

The first stage in the definition of the roof co-ordinates was the starter grid shown in black on figure D2.1a. The program that produced it is given in Appendix D3.1. This initial grid consists of four diagonal lines originating at the centre of the Reading Room, which meet the four corners of the Museum Court. The space in between is then filled with radiating lines that link evenly spaced points around the centre ring with an equal number of evenly spaced points around the rectangular perimeter. Tangential lines complete the subdivisions, resulting in a grid of trapezoidal units.

The black grid on figure D2.1a is a purely geometrical construction, the actual member grid is shown in red and this is obtained by joining appropriate nodes of the black grid. This produces the clockwise and anticlockwise 'spirals' and the radial members which run parallel to the black grid. This 'joining the dots' procedure was done

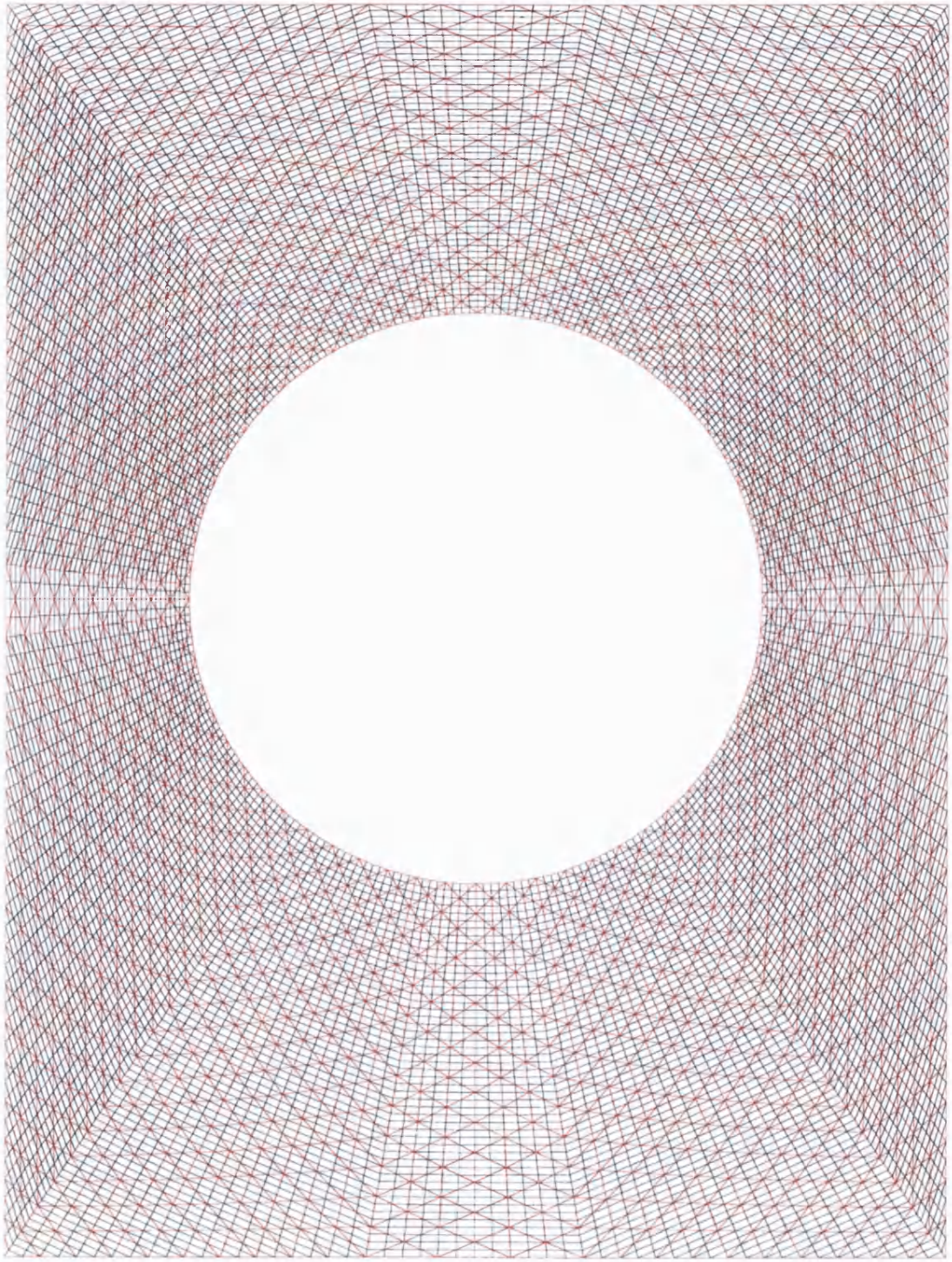


Figure D2.1a Starter grid superimposed with spiral links

by the program in appendix D3.2. A numbering system developed for the nodes made it possible to construct the spiral links by prescribing conditions relating to the numbering system. If the conditions were met, the program generated a diagonal link to produce the spiral geometry.

All nodes were then given a z value to lift the roof into shape according to the procedures described in the section entitled 'Determining the Analytical Surface'. The height of the roof was dictated by the planning height restrictions, the requirements for clearance to the North Portico and other structural and aesthetic considerations.

From figure D2.1a, it is clear that the kinks occurring throughout the spiral grid were unsatisfactory, and had to be eliminated. This was achieved using a relaxation method, which effectively moved each co-ordinate of the grid so that its x, y and z co-ordinates were the weighted average of the four surrounding nodes. However, before moving a node, the component of displacement normal to the surface (see figure D2.1b) was removed so that the node remained on the surface. This procedure is described in more detail in the section entitled 'Relaxation Procedure'.

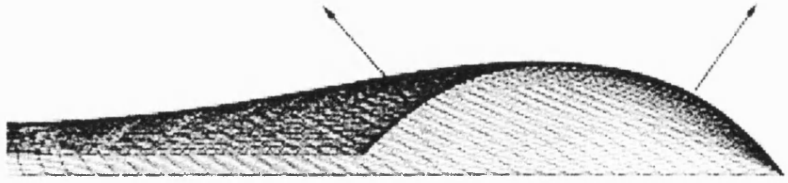


Figure D2.1b Vectors normal to surface

The process was speeded up using *dynamic relaxation*, which effectively allows all nodes to build up a velocity, thus moving them more quickly towards the desired solution. The dynamic relaxation method is discussed in more detail later

The resulting grid, in which all kinks and discontinuities were eliminated, produced a smoother grid, as can be seen from figure D2.1c.

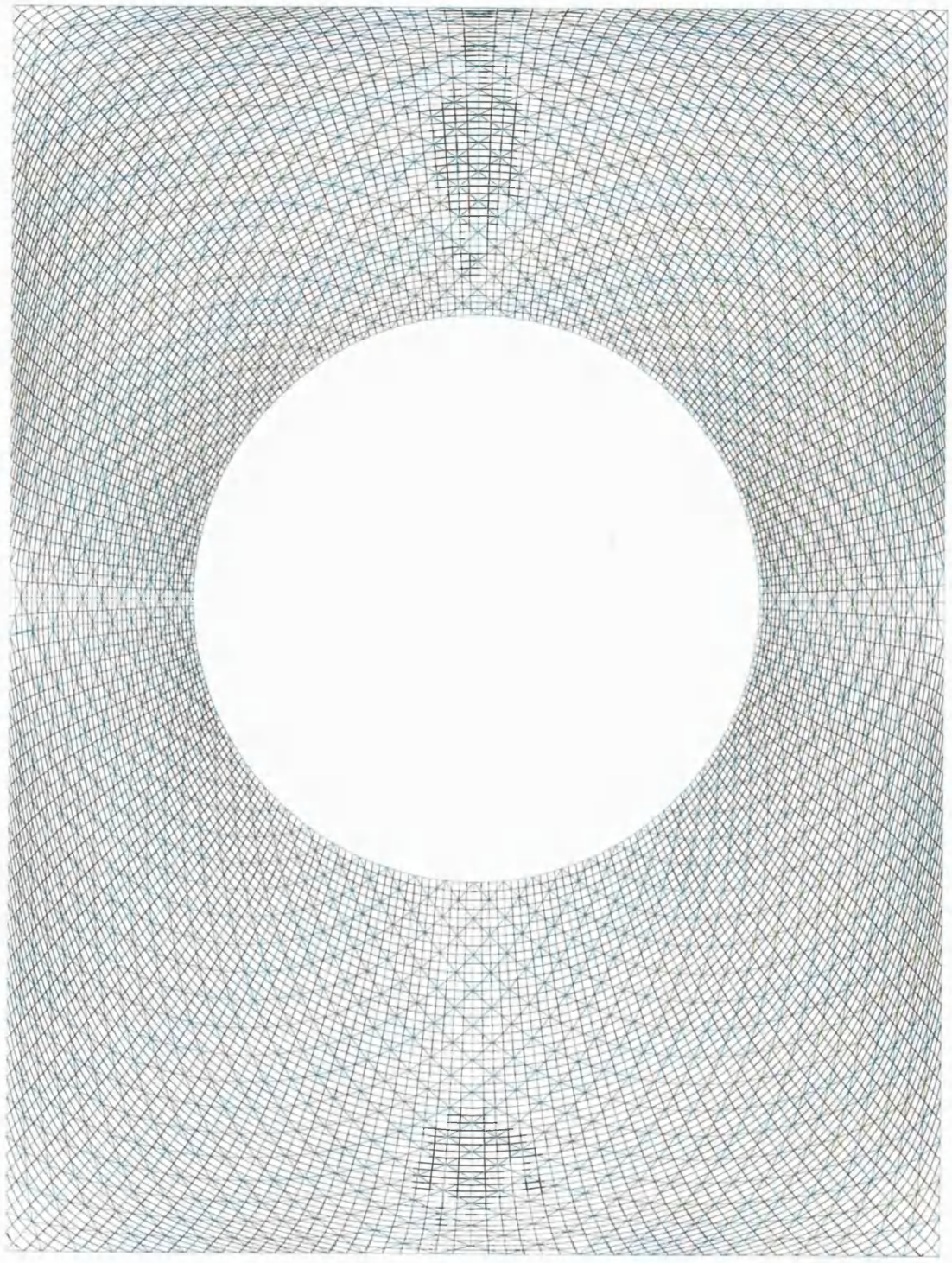


Figure D2.1c Refined grid superimposed with spiral links

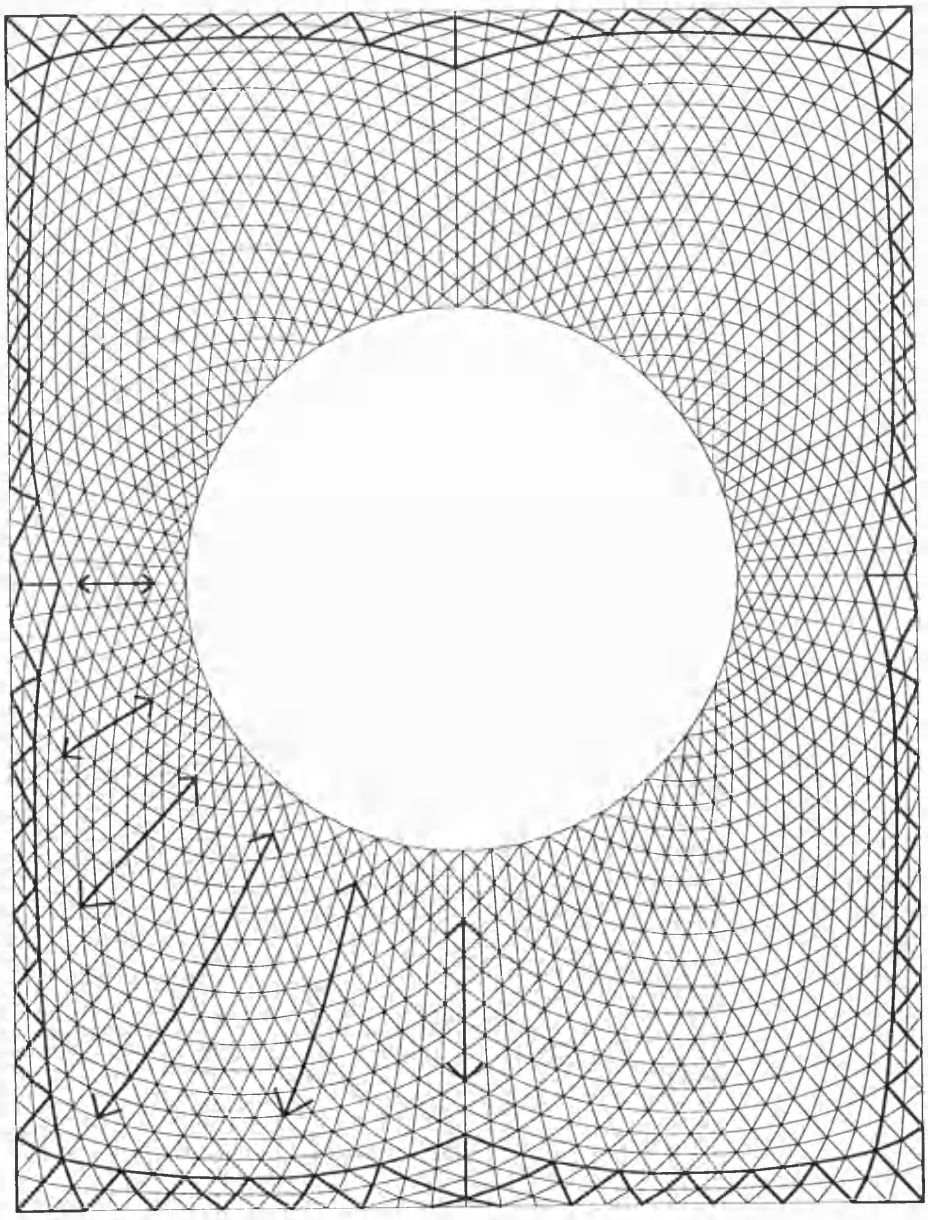


Figure D1.2b Roof layout showing structural diagram

Determining the Analytical Surface

The description of the analytical surface defining the shape of the shell involved developing a series of analytical relationships, the basis of which is given in figure D2.1d. This stage of the shape-finding process is discussed in steps 1, 2 and 3 in which

$$\left.\begin{array}{l} x = \pm b \text{ or } y = c \text{ or } y = -d \text{ around the perimeter} \\ \text{and} \\ r = \sqrt{x^2 + y^2} = a \text{ around the reading room} \end{array}\right\} \qquad (1)$$

where a, b, c and d are constant lengths.

Step 1: Lifting the centre ring by 1.2m

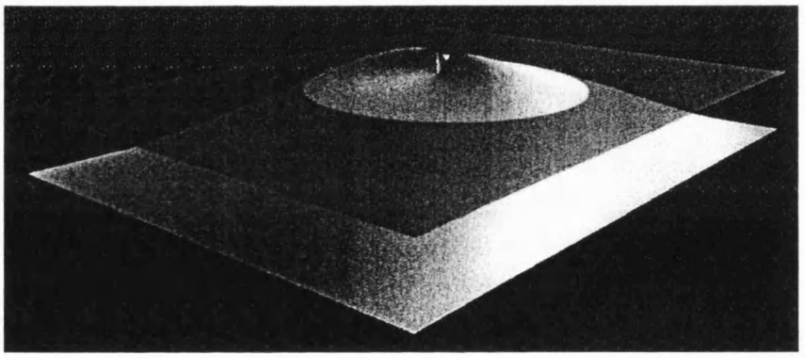


Figure D2.1e Axonometric of the initial roof surface with the centre ring lifted

$$z = H \frac{\left(1 - \frac{x}{b}\right)}{\left(1 - \frac{ax}{rb}\right)} \frac{\left(1 + \frac{x}{b}\right)}{\left(1 + \frac{ax}{rb}\right)} \frac{\left(1 - \frac{y}{c}\right)}{\left(1 - \frac{ay}{rc}\right)} \frac{\left(1 + \frac{y}{d}\right)}{\left(1 + \frac{ay}{rd}\right)} \qquad (2)$$

The relationship in D2.1(2) gives $z = 0$ around the perimeter of the roof and $z = H$ when $r = a$, around the Reading Room. H is equal to 1.2m. The effect of this relationship is illustrated in figure D2.1e.

Step 2: Creating the basic curved surface

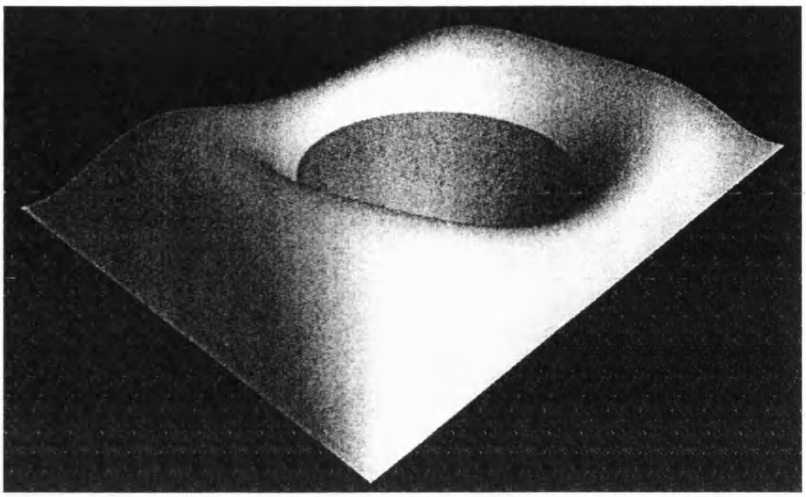


Figure D2.1f Axonometric of the curved roof showing slumped corners

$$z = H\left(1-\frac{x}{b}\right)\left(1+\frac{x}{b}\right)\left(1-\frac{y}{c}\right)\left(1+\frac{y}{d}\right)\left(\frac{\sqrt{x^2+y^2}}{a}-1\right). \qquad (3)$$

The effect produced by the relationship in D2.1(3) is shown in figure D2.1f, in which the height of the perimeter and centre ring is maintained at zero. This function gives the main form of the roof and the value of H determines the vertical scaling. In fact the value of H was also made a function of r and the plan angle $\theta = \tan^{-1}\left(\frac{y}{x}\right)$ to give further control of the height to fulfil planning and clearance requirements. The figure was plotted with H = constant.

It is clear from figure D2.1f, however, that an undesirable flattening of the roof occurs at the corners.

Step 3: Coning the roof at the corner conditions

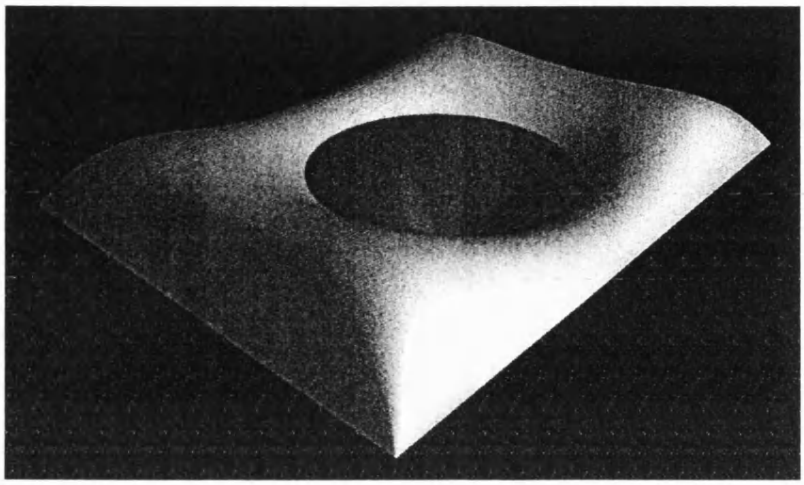


Figure D2.1g Axonometric of the curved roof surface with coned corners

$$z = \lambda \frac{\dfrac{\sqrt{x^2 + y^2}}{a} - 1}{\left(\dfrac{\sqrt{(b-x)^2 + (c-y)^2}}{(b-x)(c-y)} + \dfrac{\sqrt{(b+x)^2 + (c-y)^2}}{(b+x)(c-y)} + \dfrac{\sqrt{(b-x)^2 + (d+y)^2}}{(b-x)(d+y)} + \dfrac{\sqrt{(b+x)^2 + (d+y)^2}}{(b+x)(d+y)} \right)} \qquad (4)$$

The relationship in D2.1(4) produces the diagram in figure D2.1g in which the corners are coned. Setting $x = \pm b$, $y = c$ or $y = -d$, gives $z = 0$, at the Court perimeter, and setting $\sqrt{x^2 + y^2} = a$ again gives $z = 0$ at the centre ring. The dimensionless factor λ, is itself a function of r and the plan angle θ. Figure D2.1g was plotted for the case when λ = constant.

In order to demonstrate the effect of the relationship in D2.1(4), we shall consider the case in which x and y are very close to the corner, so that, $x = b - \varepsilon$ and $y = c - \delta$ where ε and δ are small,

$$z = \lambda \frac{\dfrac{\sqrt{(b-\varepsilon)^2+(c-\delta)^2}}{a} - 1}{\left(\dfrac{\sqrt{\varepsilon^2+\delta^2}}{\varepsilon\delta} + \dfrac{\sqrt{(2b-\varepsilon)^2+\delta^2}}{(2b-\varepsilon)\delta} + \dfrac{\sqrt{\varepsilon^2+(2d-\delta)^2}}{\varepsilon(2d-\delta)} + \dfrac{\sqrt{(2b-\varepsilon)^2+(2d-\delta)^2}}{(2b-\varepsilon)(2d-\delta)} \right)}. \quad (5)$$

Writing $\varepsilon = R\cos\beta$ and $\delta = R\sin\beta$, then as $R \to 0$,

$$z = \lambda \frac{\dfrac{\sqrt{b^2+c^2}}{a} - 1}{\dfrac{1}{R\cos\beta\sin\beta} + \dfrac{1}{R\sin\beta} + \dfrac{1}{R\cos\beta}} \quad (6)$$

or

$$z = \lambda \frac{R\left(\dfrac{\sqrt{b^2+c^2}}{a} - 1\right)\cos\beta\sin\beta}{1+\cos\beta+\sin\beta}. \quad (7)$$

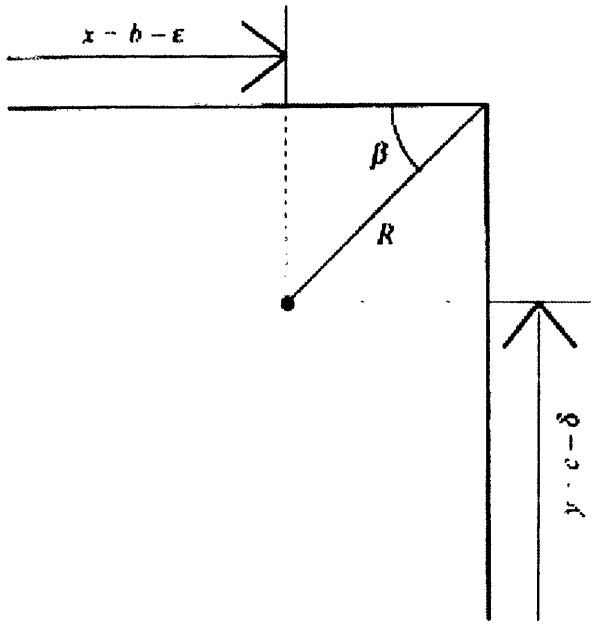


Figure D2.1h Diagram of a typical node near the roof corner

R and β shown in figure D2.1h and this, together with D2.1(7) shows how a conical corner is produced. At the corner itself the curvature of the surface is infinite, like the point of a cone.

Relaxation Procedure

Following the definition of the correct surface, the procedure below describes the method used to adjust the spacing of nodes on the surface in order to refine the spiral pattern.

Let us consider the x and y co-ordinates of a certain node, (i, j), and its relationship to the co-ordinates of the four nodes, $(i+1, j)$, $(i-1, j)$, $(i, j+1)$ and $(i, j-1)$ surrounding it as shown in figure D2.1i. Note that this grid is the black grid in figure D2.1a, **not** the real member grid shown in red on the figure.

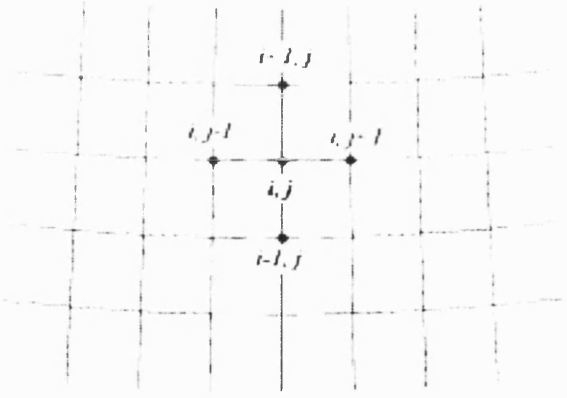


Figure D2.1i A typical node, i, j, with four surrounding nodes

The x co-ordinate of the node (i, j) can be expressed in terms of a series of linear simultaneous equations given by,

$$x_{i,j} = \frac{1}{4}\left(\lambda x_{i+1,j} + (2-\lambda)x_{i-1,j} + x_{i,j+1} + x_{i,j-1}\right), \qquad (8)$$

in which, λ, is a weighting factor used to control and adjust the spacing of nodes. The weighting factor is varied to control member lengths as the Reading Room is approached

and to limit the size of glass panels at the outer perimeter. Note that the sum of the weights in D2.1(8) is equal to $\frac{1}{4}\left(\lambda+\left(2-\lambda\right)+1+1\right)=1$ as one would expect.

The equation in D2.1(8) can be solved using various techniques, including matrix methods, which is the standard way for solving linear simultaneous equations. In this case, however, a relaxation process was used.

The simplest relaxation process is to repeatedly apply the averaging process to each node in turn going through the whole structure a large number of times. This iteration procedure has to be repeated since in moving any one node, the surrounding nodes are affected. As mentioned in the introductory part of this section, this process was speeded up using *Dynamic Relaxation*, a technique first proposed by Day (1965) in his work concerning tidal flow with Professor Otter.

Barnes (1977), Wakefield (1980), Papadrakis (1978) and Topping (1978) give fuller descriptions of DR as applied to cable nets, tension and space structures.

Before concluding this study, it is worth pointing out that the desired configuration of the steel grid was not produced after the first, or even second attempt. The procedure to arrive at the final solution involved a process of trial and error in which adjustments were continually incorporated into the program until the desired result was achieved.

The geometrical analysis was developed in conjunction with the structural analysis of the roof. The numerical data generated by the geometrical analysis provided the input for the non-linear structural analysis.

Feedback from the structural iterations sometimes meant adjustments to the geometry and geometrical constraints sometimes meant that the structural assumptions had to be re-considered.

The program to generate the grid the initial starting grid and then perform the relaxation procedure, is given in Appendix D3.0.

D2.2 Geometrical Analysis of the Glass Panels

The original proposal was for a cluster of four single glazed triangular panels supported on a single triangular steel grid module with point supports. However, the need to satisfy more onerous thermal constraints called for double-glazing and Foster and Partners did not want the dark lines caused by the glazing joints to be exposed. This lead to a change to a finer structural grid so that only one double-glazed unit is supported on each triangular steel grid module .

A steel grid was chosen to conform to the maximum height and length of a sealed double-glazed unit, which measures 2.2m x 3.5m. Therefore, although the geometry of the steel structure dictated the geometry of the glass skin, the manufacturing possibilities of the glass determined the coarseness of the steel grid.

The System Point Geometry sets out the geometry of the top surface of the steel assembly, and an offset above it designed to accommodate the zone of the double-glazed unit determines the profile of the glass skin. This information was issued to the design team in the form of a data file, which contained a table giving information about the node number, the x, y and z co-ordinates of its system point, the angle θ and the angle ϕ in degrees, where θ and ϕ are the angles of the surface normal according to the diagram in figure D2.2a.

Numerical data generated by the computer program listing dimensional and geometric characteristics of panels made it possible to check compliance of panel dimensions against the limitations imposed by the manufacturing process of the glass.
This process helped to rule out poorly proportioned panels and panels with acute internal angles. The latter criterion was imposed firstly due to the impracticability of producing an adequate sealing detail in tight corners, which would have repercussions on the node geometry in terms of accessibility for welding and secondly, to avoid susceptibility to damage of glass units.

The author's involvement at this stage of the project was to produce a short computer program that applied sorting criteria to the glass panels in order to establish whether or not repetition existed and therefore the extent to which inter-changeability and modular production of the glass units was possible.

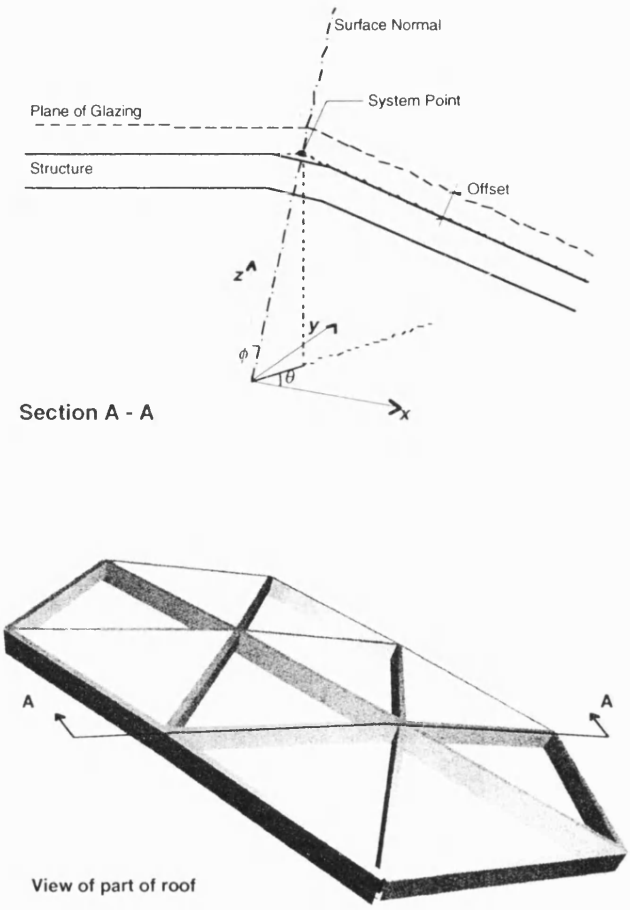


Figure D2.2a System point geometry of a given node

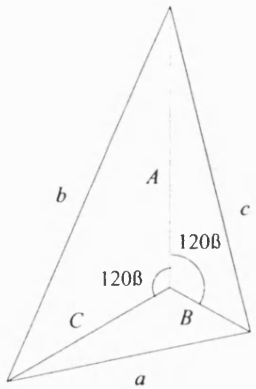


Figure D2.2b Typical glass panel

Figure D2.2b shows a typical triangular glass panel positioned and rotated so that each of its corners lie on one of three lines at 120° to each other. The reason for doing this is to facilitate superimposing panels to see how near they are in shape and size.

The distances a, b and c can be readily calculated from the Cartesian co-ordinates of the corners using Pythagoras' theorem. The distances A, B and C can be calculated from a, b and c by first using the cosine rule on each of the sub-triangles:

$$\begin{aligned} a^2 &= B^2 + C^2 - 2BC\cos 120^\circ \\ &= B^2 + C^2 + BC, \\ b^2 &= C^2 + A^2 + CA \\ &\text{and} \\ c^2 &= A^2 + B^2 + AB. \end{aligned}$$

These three equations can be solved to give

$$A = \frac{Q^2 - a^2 + b^2 + c^2}{\sqrt{2}\sqrt{3Q^2 + a^2 + b^2 + c^2}},\ B = \frac{Q^2 + a^2 - b^2 + c^2}{\sqrt{2}\sqrt{3Q^2 + a^2 + b^2 + c^2}} \text{ and}$$

$$C = \frac{Q^2 + a^2 + b^2 - c^2}{\sqrt{2}\sqrt{3Q^2 + a^2 + b^2 + c^2}}$$

$$\text{where } Q^2 = \sqrt{\frac{2\left(b^2c^2 + c^2a^2 + a^2b^2\right) - \left(a^4 + b^4 + c^4\right)}{3}}.$$

That these are indeed the solution can be checked by back substitution,

$$A^2+B^2+AB=\frac{3(A+B)^2+(A-B)^2}{4}=\frac{12\left(Q^2+c^2\right)^2+4\left(-a^2+b^2\right)^2}{4\times 2\left(3Q^2+a^2+b^2+c^2\right)}$$

$$=\frac{3\left(Q^4+2Q^2c^2+c^4\right)+a^4-2a^2b^2+b^4}{2\left(3Q^2+a^2+b^2+c^2\right)}$$

$$=\frac{2\left(b^2c^2+c^2a^2+a^2b^2\right)-\left(a^4+b^4+c^4\right)+6Q^2c^2+3c^4+a^4-2a^2b^2+b^4}{2\left(3Q^2+a^2+b^2+c^2\right)}$$

$$=\frac{2\left(b^2c^2+c^2a^2\right)+6Q^2c^2+2c^4}{2\left(3Q^2+a^2+b^2+c^2\right)}$$

$$=c^2.$$

The area of the triangle is equal to $\frac{1}{2}(BC+CA+AB)\sin 60^\circ=\frac{\sqrt{3}}{4}(BC+CA+AB)$

and it can be shown that this is equal to $\frac{\sqrt{3}}{4}Q^2$.

Each triangle has three degrees of freedom, a, b or A, B and C. Alternatively the three degrees of freedom can be expressed by

$$\text{an Equilateral measure}=\frac{\sqrt{(B-C)^2+(C-A)^2+(A-B)^2}}{\sqrt{A^2+B^2+C^2}},$$

$$\text{an Isosceles measure}=\frac{\left[(B-C)(C-A)(A-B)\right]^{\frac{1}{3}}}{\sqrt{A^2+B^2+C^2}}$$

and the size of the triangle. The equilateral measure and isosceles measure are non-dimensional and they express the *shape* of the triangle. The equilateral measure is zero if the triangle is equilateral and the isosceles measure is zero if the triangle is isosceles measure. If two triangles have the same equilateral measures and the isosceles measures, then the two triangles are similar (that is they have the same shape, but differ only in size). This only true if a sign is given to the equilateral measure to distinguish between 'tall thin' and 'short fat' triangles.

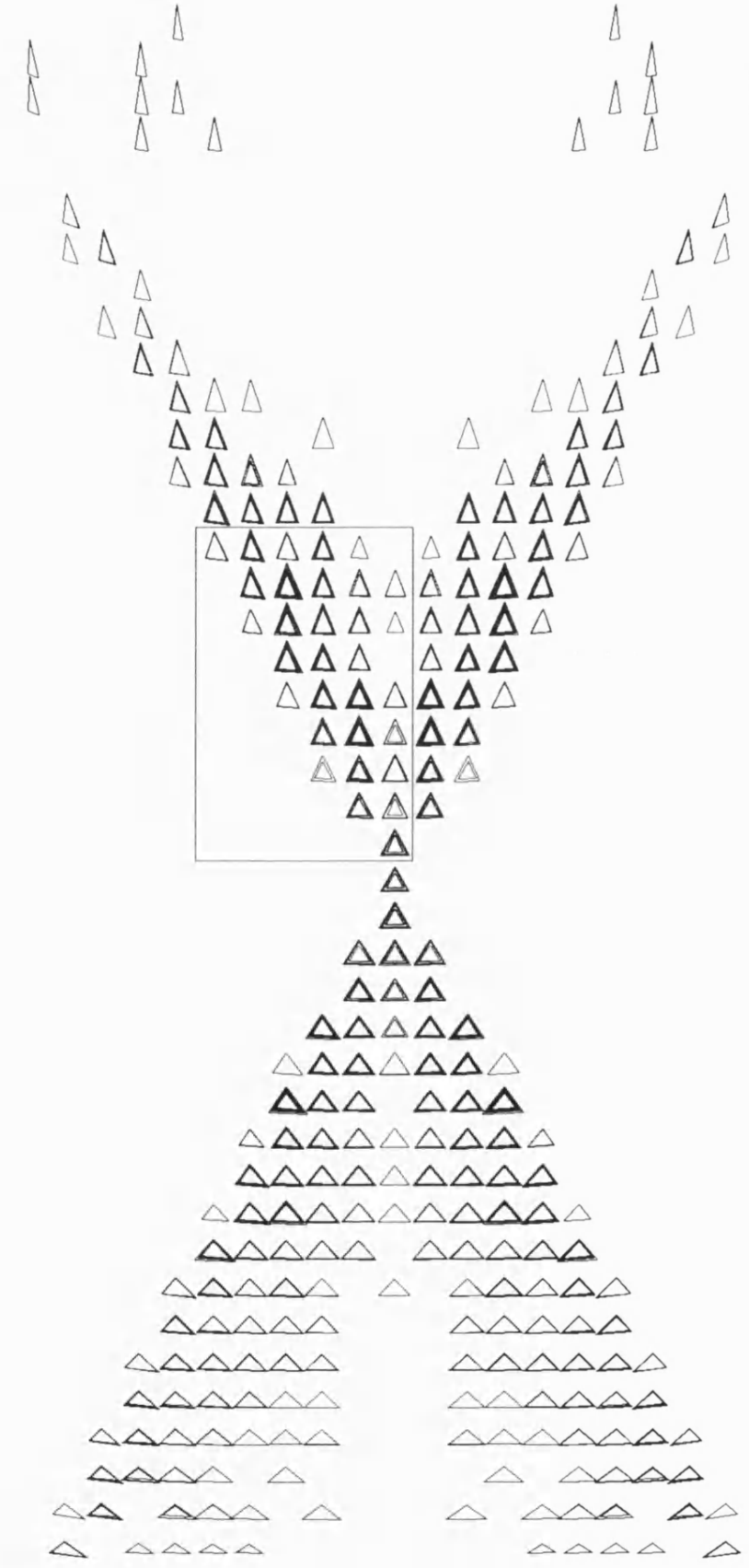


Figure D2.2c Sorting of glass panels

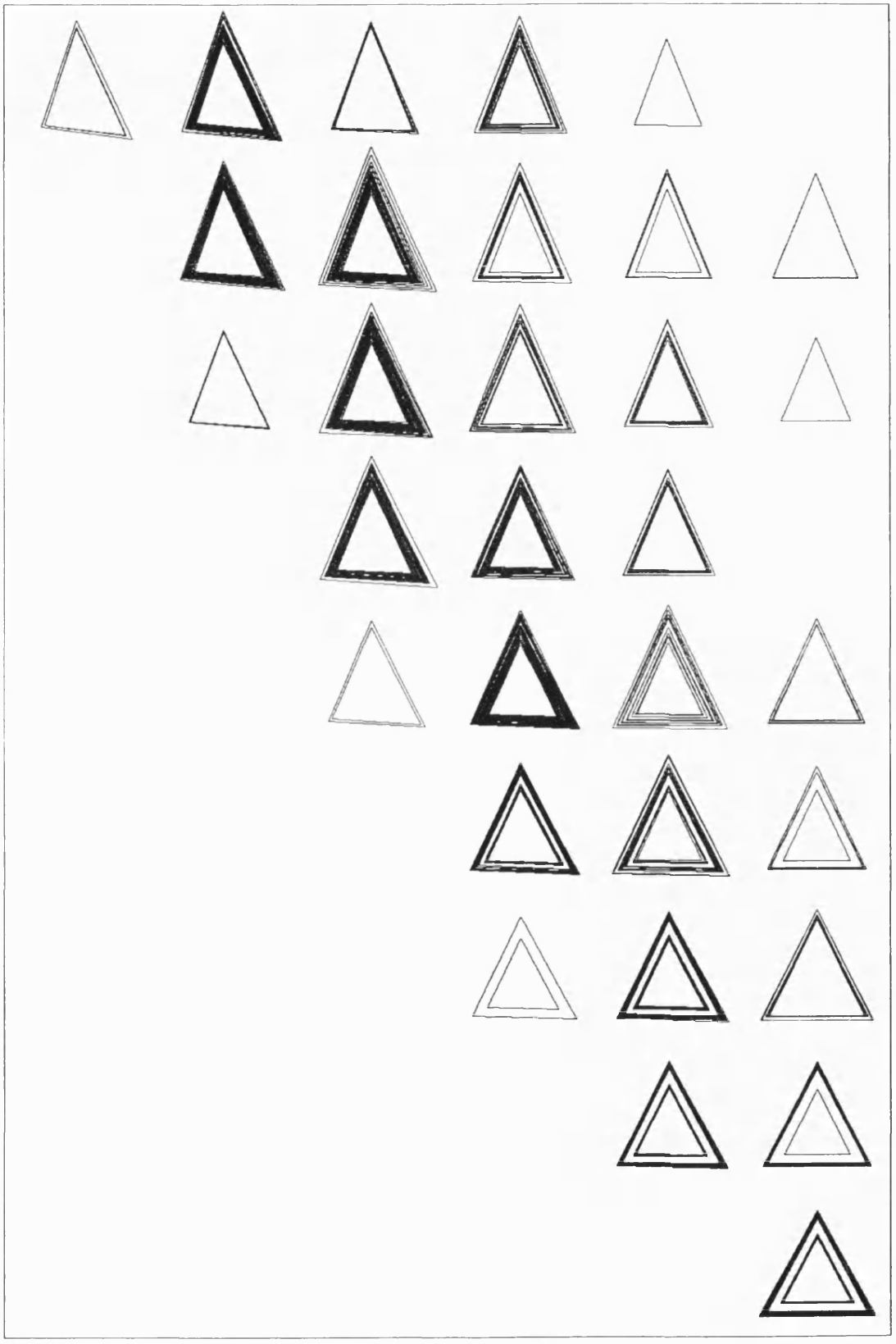


Figure D2.2d Enlarged area

The result of this sorting process is shown in figures D2.2c and D2.2d in which the area within the rectangle in D2.2c is enlarged. The vertical position of each triangle depends upon the equilateral measure and the horizontal position depends upon the isosceles measure so that triangles on the vertical axis of symmetry are isosceles or very nearly isosceles. The diagram is symmetric since the roof itself is symmetric so that for every triangle there is its mirror image - except for triangles on the line of symmetry which must be isosceles.

This drawing was produced by the following computer program:

```
#include <math.h>
#include <stdio.h>
#define   nodenum      2500
#define   membernum    6000
#define   panenum      4000
#define   glassabovesystem 0.025

void Rotate(void);

void dxf3DPaneFace(void);

float PI,x[nodenum],y[nodenum],z[nodenum],ytoplot[nodenum],
      polartheta,polarphi,

glassx1,glassy1,glassz1,glassx2,glassy2,glassz2,glassx3,glassy3,glassz
3,
      xnormal,ynormal,znormal,
      PaneNormx,PaneNormy,PaneNormz,
      asq,bsq,csq,
      temp1,temp2,areax4,r3,A[panenum],B[panenum],C[panenum],
      EquiMeasure,IsosMeasure,Motion,
      xshift,yshift;

int   nodeno,PaneNo,memberno,totalnodeno,totalmemberno,
      totalpaneno,tempint,
      Node1[panenum],Node2[panenum],Node3[panenum],
      EquiFraction,IsosFraction;

FILE  *DataFile,*Pane;

void main(void)
{
EquiFraction=20;
IsosFraction=20;
Motion=3.0;
r3=sqrt(3.0);
PI=4.0*atan(1.0);

DataFile=fopen("Numbers","r");
fscanf(DataFile,"%d%d%d",&totalnodeno,&totalmemberno,&totalpaneno);
fclose(DataFile);

DataFile=fopen("SystemPoints","r");
for(tempint=1;tempint<=totalnodeno;tempint+=1)
{
fscanf(DataFile,"%d",&nodeno);
fscanf(DataFile,"%f%f%f%f%f",&x[nodeno],&y[nodeno],&z[nodeno],&polarth
eta,&polarphi);
```

```
polartheta=polartheta*PI/180.0;
polarphi=polarphi*PI/180.0;
xnormal=sin(polarphi)*cos(polartheta);
ynormal=sin(polarphi)*sin(polartheta);
znormal=cos(polarphi);
x[nodeno]+=xnormal*glassabovesystem;
y[nodeno]+=ynormal*glassabovesystem;
z[nodeno]+=znormal*glassabovesystem;
}
fclose(DataFile);

DataFile=fopen("TriangleNodes","r");
for(tempint=1;tempint<=totalpaneno;tempint+=1)
{
fscanf(DataFile,"%d",&PaneNo);
fscanf(DataFile,"%d%d%d",&Node1[PaneNo],&Node2[PaneNo],&Node3[PaneNo])
;
}
fclose(DataFile);

for(PaneNo=1;PaneNo<=totalpaneno;PaneNo+=1)
{
PaneNormz=(x[Node2[PaneNo]]-x[Node1[PaneNo]])*(y[Node3[PaneNo]]-
y[Node1[PaneNo]])
          -(x[Node3[PaneNo]]-x[Node1[PaneNo]])*(y[Node2[PaneNo]]-
y[Node1[PaneNo]]);
if(PaneNormz<0.0)
{
tempint=Node2[PaneNo];
Node2[PaneNo]=Node3[PaneNo];
Node3[PaneNo]=tempint;
}
}


for(PaneNo=1;PaneNo<=totalpaneno;PaneNo+=1)
{
asq=(x[Node3[PaneNo]]-x[Node2[PaneNo]])*(x[Node3[PaneNo]]-
x[Node2[PaneNo]])
   +(y[Node3[PaneNo]]-y[Node2[PaneNo]])*(y[Node3[PaneNo]]-
y[Node2[PaneNo]])
   +(z[Node3[PaneNo]]-z[Node2[PaneNo]])*(z[Node3[PaneNo]]-
z[Node2[PaneNo]]);
bsq=(x[Node1[PaneNo]]-x[Node3[PaneNo]])*(x[Node1[PaneNo]]-
x[Node3[PaneNo]])
   +(y[Node1[PaneNo]]-y[Node3[PaneNo]])*(y[Node1[PaneNo]]-
y[Node3[PaneNo]])
   +(z[Node1[PaneNo]]-z[Node3[PaneNo]])*(z[Node1[PaneNo]]-
z[Node3[PaneNo]]);
csq=(x[Node2[PaneNo]]-x[Node1[PaneNo]])*(x[Node2[PaneNo]]-
x[Node1[PaneNo]])
   +(y[Node2[PaneNo]]-y[Node1[PaneNo]])*(y[Node2[PaneNo]]-
y[Node1[PaneNo]])
   +(z[Node2[PaneNo]]-z[Node1[PaneNo]])*(z[Node2[PaneNo]]-
z[Node1[PaneNo]]);
areax4=sqrt(2.0*(asq*bsq+bsq*csq+csq*asq)-(asq*asq+bsq*bsq+csq*csq));

temp1=areax4/r3+asq+bsq+csq;
temp2=sqrt(2.0*r3*areax4+2.0*(asq+bsq+csq));
A[PaneNo]=(temp1-2.0*asq)/temp2;
B[PaneNo]=(temp1-2.0*bsq)/temp2;
C[PaneNo]=(temp1-2.0*csq)/temp2;

Rotate();
Rotate();
```

```
}

Pane=fopen("PaneSort.dxf","w");
fprintf(Pane,"0\r");
fprintf(Pane,"SECTION\r");
fprintf(Pane,"2\r");
fprintf(Pane,"ENTITIES\r");

for(PaneNo=1;PaneNo<=totalpaneno;PaneNo+=1)
{

temp1=(B[PaneNo]-C[PaneNo])*(C[PaneNo]-A[PaneNo])*(A[PaneNo]-
B[PaneNo]);
IsosMeasure=pow(temp1*temp1,1.0/6.0)/
                //This is because pow does not work on negative numbers
sqrt(A[PaneNo]*A[PaneNo]+B[PaneNo]*B[PaneNo]+C[PaneNo]*C[PaneNo]);
if(temp1<0.0)IsosMeasure=-IsosMeasure;
xshift=Motion*(int)(IsosFraction*IsosMeasure);

EquiMeasure=sqrt(((B[PaneNo]-C[PaneNo])*(B[PaneNo]-C[PaneNo])
                  +(C[PaneNo]-A[PaneNo])*(C[PaneNo]-A[PaneNo])
                  +(A[PaneNo]-B[PaneNo])*(A[PaneNo]-B[PaneNo]))/
     (A[PaneNo]*A[PaneNo]+B[PaneNo]*B[PaneNo]+C[PaneNo]*C[PaneNo]));
if(A[PaneNo]<(B[PaneNo]+C[PaneNo])/2.0)EquiMeasure=-EquiMeasure;
yshift=Motion*(int)(EquiFraction*EquiMeasure);

glassx1=xshift;glassy1=A[PaneNo]+yshift;
glassx2=B[PaneNo]*r3/2.0+xshift;glassy2=-B[PaneNo]/2.0+yshift;
glassx3=-C[PaneNo]*r3/2.0+xshift;glassy3=-C[PaneNo]/2.0+yshift;
dxf3DPaneFace();
}

fprintf(Pane,"0\r");
fprintf(Pane,"ENDSEC\r");
fprintf(Pane,"0\r");
fprintf(Pane,"EOF\r");
fclose(Pane);
}

void Rotate(void)
{
if((asq-bsq)*(asq-bsq)<(bsq-csq)*(bsq-csq)||(csq-asq)*(csq-asq)<(bsq-
csq)*(bsq-csq))
{
temp1=A[PaneNo];
A[PaneNo]=B[PaneNo];
B[PaneNo]=C[PaneNo];
C[PaneNo]=temp1;

temp2=asq;
asq=bsq;
bsq=csq;
csq=temp2;
}
}

void dxf3DPaneFace(void)
{
fprintf(Pane,"0\rPOLYLINE\r8\r0\r");
fprintf(Pane,"62\r0\r70\r1\r");
fprintf(Pane,"0\rVERTEX\r8\r0\r");
```

```
fprintf(Pane,"62\r0\r70\r1\r");
fprintf(Pane,"10\r%f\r",glassx1);
fprintf(Pane,"20\r%f\r",glassy1);
fprintf(Pane,"0\rVERTEX\r8\r0\r");
fprintf(Pane,"62\r0\r70\r1\r");
fprintf(Pane,"10\r%f\r",glassx2);
fprintf(Pane,"20\r%f\r",glassy2);
fprintf(Pane,"0\rVERTEX\r8\r0\r");
fprintf(Pane,"62\r0\r70\r1\r");
fprintf(Pane,"10\r%f\r",glassx3);
fprintf(Pane,"20\r%f\r",glassy3);
fprintf(Pane,"0\rSEQEND\r");
}
```

The position of each triangle is decided by the portions of program in boxes in which the equilateral measure and isosceles measure are calculated. The positions of the triangles are stepped by integer values of (EquiFraction*EquiMeasure) and (IsosFraction*IsosMeasure) in which the EquiFraction and IsosFraction control the amount of variation of the equilateral measure and isosceles measure for each triangle location. The subroutine Rotate is to rotate the triangles so that the largest of the quantities *A*, *B* and *C* is on the vertical leg of the three 120° lines.

Examination of drawings like figure D2.2c with varying values of EquiFraction and IsosFraction showed that very few panels were of sufficiently similar size and shape to warrant using modular panels, especially since tolerances on every aspect of the steel and glass were very tight. However, since the glass panels were made using CNC equipment, this does not seem to have had any significant cost implications.

D3.0 British Museum Roof Computer Programs

D3.1 British Museum Roof computer program to generate starting grid and perform dynamic relaxation

```
#include <math.h>
#include <stdio.h>

#define   surfacenum  36000
#define   linenum       400
#define   maxnumrowsp1  801
#define   maxnocols     400
#define   zrange      100.0
#define   delta         0.01
#define   centreheight 20.955
#define   edgeheight   19.71
#define   normalcalc    1

extern void SetUpDrawingArea(int halfwidth, int halfheight);
extern void FinishOffDrawing(void);
//extern void DrawLinesAndSurface(int nlines,short int linetype[],
//                                    float xline[],float yline[],float
zline[],
//                                int m,int n,
//                                    float xsurf[],float ysurf[],float
zsurf[]);

void setup(void);
void setupxy(void);
void dothebound(void);
void setbound(void);
void typicalnodexy(void);
void grid(void);
void circumference(void);
void polar(void);
void calcxy(void);
void calczanalytic(void);
void findtheheight(void);
void findnormals(void);

double PI,

tempfloat,tempfloatread1,tempfloatread2,tempfloatread3,tempfloatread4,
tempfloatread5,
       xline[linenum],yline[linenum],zline[linenum],
       x[surfacenum],y[surfacenum],z[surfacenum],
       ytoplot[surfacenum],
       dzbydx[surfacenum],dzbydy[surfacenum],

xmovement[surfacenum],ymovement[surfacenum],zmovement[surfacenum],

prevxmovement[surfacenum],prevymovement[surfacenum],prevzmovement[surf
acenum],
       rotation1,rotation2,scale,crot1,srot1,crot2,srot2,
       tx,ty,tz,
       a,perpdist[4],r,theta,
       ratio,
       xdistance,ydistance,grad,ConeAmount,
       xvalue,yvalue,zvalue,storezvalue[3][3],
       xnormal[surfacenum],ynormal[surfacenum],znormal[surfacenum],
```

```
        lambda,mu,
        previousKE,KE,overrelax,carryover,radialx,radialy,radialz,
        Waagner;

int     step,row,col,numrows,numcols,
        topcol[4],botcol[4],toprow[4],botrow[4],minbotrow,maxtoprow,
        boundaryrow[maxnumrowsp1],
        minnumrows,
        rowstodraw,colstodraw,colstodrawperside[4],noddy,Q,
        k,
        whichcol,tempint,
        row2,col2,row3,col3,
        phiorw,temprow,trow,tcol,
        cycle,cyclesSoFar,maxnumxycycles,
        wherex,wherey,
        tempintread1,tempintread2,tempintread3,calcxyornot,
        intermediatecycle,intermediate,
        intxdraw,intydraw;

FILE   *DataFile,*OutPutFile;

short int linetype[linenum];

void main(void)
{
for(;;)
{
printf("0 = start from original coords and do not change x & y\n");
printf("1 = start from original coords and do change x & y\n");
printf("2 = start from scratch\n");
scanf("%d",&calcxyornot);
if(calcxyornot==0||calcxyornot==1||calcxyornot==2)break;
}

PI=4.0*atan(1.0);
step=2;

DataFile=fopen("DimData","r");
fscanf(DataFile,"%lf",&tempfloat);a=tempfloat;
for(k=0;k<=3;k+=1){fscanf(DataFile,"%lf",&tempfloat);perpdist[k]=tempf
loat;}
fclose(DataFile);

DataFile=fopen("TopologyData","r");
fscanf(DataFile,"%d",&minnumrows);
for(k=0;k<=3;k+=1)fscanf(DataFile,"%d",&topcol[k]);
for(k=0;k<=2;k+=1)fscanf(DataFile,"%d",&botcol[k]);
fclose(DataFile);

setup();

Q=numcols+1;

if(calcxyornot==0||calcxyornot==1)
{
DataFile=fopen("Coordinates","r");
fscanf(DataFile,"%d%d",&tempintread1,&tempintread2);
for(k=0;k<=3;k+=1)fscanf(DataFile,"%d",&tempintread1);
for(k=0;k<=3;k+=1)fscanf(DataFile,"%d",&tempintread1);
for(row=0;row<=numrows;row+=1)
{
for(col=0;col<=numcols;col+=1)
{
fscanf(DataFile,"%d%d%d%le%le%le%le%le",
  &tempintread1,&tempintread2,
```

```
  &tempintread3,
  &tempfloatread1,
  &tempfloatread2,
  &tempfloatread3,
  &tempfloatread4,
 *&tempfloatread5);
x[Q*row+col]=tempfloatread1;
y[Q*row+col]=tempfloatread2;
}
}
}
fclose(DataFile);
}

if(calcxyornot==1)calcxy();

if(calcxyornot==2)
{
setupxy();
calcxy();
}

calczanalytic();

if(perpdist[0]==perpdist[2])
{
for(row=0;row<=numrows;row+=1)
{
for(col=0;col<=numcols;col+=1)
{
if(col<=botcol[2])
{
     x[Q*row+botcol[2]-col]=-x[Q*row+col];
     y[Q*row+botcol[2]-col]=+y[Q*row+col];
     z[Q*row+botcol[2]-col]=+z[Q*row+col];
dzbydx[Q*row+botcol[2]-col]=-dzbydx[Q*row+col];
dzbydy[Q*row+botcol[2]-col]=dzbydy[Q*row+col];
}
if(col>botcol[2])
{
     x[Q*row+numcols+botcol[2]-col]=-x[Q*row+col];
     y[Q*row+numcols+botcol[2]-col]=+y[Q*row+col];
     z[Q*row+numcols+botcol[2]-col]=+z[Q*row+col];
dzbydx[Q*row+numcols+botcol[2]-col]=-dzbydx[Q*row+col];
dzbydy[Q*row+numcols+botcol[2]-col]=dzbydy[Q*row+col];
}
}
}
}
}

OutPutFile=fopen("Coordinates","w");
fprintf(OutPutFile,"%d  %d\r",numrows,numcols);
for(k=0;k<=3;k+=1)fprintf(OutPutFile,"%d\r",topcol[k]);
for(k=0;k<=3;k+=1)fprintf(OutPutFile,"%d\r",botcol[k]);
for(row=0;row<=numrows;row+=1)
{
for(col=0;col<=numcols;col+=1)
fprintf(OutPutFile,"%d  %d  %d\r%le  %le  %le  %le  %le\r",
  row,col,
  boundaryrow[col],
  x[Q*row+col],
  y[Q*row+col],
  z[Q*row+col],
  dzbydx[Q*row+col],
  dzbydy[Q*row+col]);
}
```

```
fclose(OutPutFile);

scale=5.0;

rotation1=70.0;
rotation2=60.0;

crot1=cos(rotation1*PI/180.0);
srot1=sin(rotation1*PI/180.0);
crot2=cos(rotation2*PI/180.0);
srot2=sin(rotation2*PI/180.0);

for(row=0;row<=numrows;row+=1)
{
for(col=0;col<=numcols;col+=1)
{
tx=scale*x[Q*row+col];
ty=scale*y[Q*row+col];
if(z[Q*row+col]>zrange)z[Q*row+col]=zrange;
if(z[Q*row+col]<-zrange)z[Q*row+col]=-zrange;
tz=scale*z[Q*row+col];
x[Q*row+col]=tx*crot1-ty*srot1;
ytoplot[Q*row+col]=tx*srot1+ty*crot1;
y[Q*row+col]=ytoplot[Q*row+col]*crot2+tz*srot2;
z[Q*row+col]=-ytoplot[Q*row+col]*srot2+tz*crot2;
}
}

SetUpDrawingArea(317,218);
//DrawLinesAndSurface(NoLinesSoFar,linetype,xline,yline,zline,numrows,
numcols,x,y,z);
for(row=0;row<=numrows;row+=1)
{
for(col=0;col<=numcols;col+=1)
{
intxdraw=317+x[Q*row+col];
intydraw=218-y[Q*row+col];
if(col==0)MoveTo(intxdraw,intydraw);
LineTo(intxdraw,intydraw);
}
}
for(col=0;col<=numcols;col+=1)
{
for(row=0;row<=numrows;row+=1)
{
intxdraw=317+x[Q*row+col];
intydraw=218-y[Q*row+col];
if(row==0)MoveTo(intxdraw,intydraw);
LineTo(intxdraw,intydraw);
}
}
FinishOffDrawing();
printf("Finished\n");
}

void setup(void)
{
botcol[3]=2*(topcol[0]+topcol[1]+topcol[2]+topcol[3]-botcol[0]-
botcol[1]-botcol[2]);

minnumrows=step*minnumrows;
for(k=0;k<=3;k+=1)
{
topcol[k]=step*topcol[k];
botcol[k]=step*botcol[k];
```

```
}

numcols=botcol[3];

toprow[0]=0;
for(k=0;k<=3;k+=1)
{
if(k!=0)toprow[k]=botrow[k-1]+topcol[k]-botcol[k-1];
botrow[k]=toprow[k]-(botcol[k]-topcol[k]);
}

minbotrow=botrow[0];
for(k=1;k<=3;k+=1){if(botrow[k]<minbotrow)minbotrow=botrow[k];}

for(k=0;k<=3;k+=1)
{
toprow[k]-=minbotrow;
botrow[k]-=minbotrow;
}

maxtoprow=toprow[0];
for(k=1;k<=3;k+=1){if(toprow[k]>maxtoprow)maxtoprow=toprow[k];}

numrows=maxtoprow+minnumrows;

for(k=0;k<=3;k+=1)
{
if(k==0){for(col=0;col<=topcol[0]-
1;col+=1)boundaryrow[col]=botrow[3]+col;}
else
{for(col=botcol[k-1];col<=topcol[k]-
1;col+=1)boundaryrow[col]=botrow[k-1]+col-botcol[k-1];}

for(col=topcol[k];col<=botcol[k]-1;col+=1)
{
boundaryrow[col]=toprow[k]-(col-topcol[k]);
}
if(k==0||k==2)
{
boundaryrow[topcol[k]-1]-=1;
boundaryrow[topcol[k]]-=2;
boundaryrow[topcol[k]+1]-=1;
}
if(k==1||k==3)
{
boundaryrow[topcol[k]-5]-=1;
boundaryrow[topcol[k]-4]-=2;
boundaryrow[topcol[k]-3]-=3;
boundaryrow[topcol[k]-2]-=4;
boundaryrow[topcol[k]-1]-=5;
boundaryrow[topcol[k]]-=6;
boundaryrow[topcol[k]+1]-=5;
boundaryrow[topcol[k]+2]-=4;
boundaryrow[topcol[k]+3]-=3;
boundaryrow[topcol[k]+4]-=2;
boundaryrow[topcol[k]+5]-=1;
}
}
boundaryrow[numcols]=botrow[3];
}

void setupxy(void)
{
for(col=0;col<=topcol[0];col+=1)y[Q*boundaryrow[col]+col]
=((1.0*col-1.0*topcol[0])/(1.0*topcol[0]))*perpdist[3];
```

```
for(col=topcol[0]+1;col<=botcol[0];col+=1)y[Q*boundaryrow[col]+col]
=((1.0*col-1.0*topcol[0])/(1.0*botcol[0]-1.0*topcol[0]))*perpdist[1];

for(col=botcol[0]+1;col<=topcol[1];col+=1)x[Q*boundaryrow[col]+col]
=((1.0*topcol[1]-1.0*col)/(1.0*topcol[1]-1.0*botcol[0]))*perpdist[0];
for(col=topcol[1]+1;col<=botcol[1];col+=1)x[Q*boundaryrow[col]+col]
=((1.0*topcol[1]-1.0*col)/(1.0*botcol[1]-1.0*topcol[1]))*perpdist[2];

for(col=botcol[1]+1;col<=topcol[2];col+=1)y[Q*boundaryrow[col]+col]
=((1.0*topcol[2]-1.0*col)/(1.0*topcol[2]-1.0*botcol[1]))*perpdist[1];
for(col=topcol[2]+1;col<=botcol[2];col+=1)y[Q*boundaryrow[col]+col]
=((1.0*topcol[2]-1.0*col)/(1.0*botcol[2]-1.0*topcol[2]))*perpdist[3];

for(col=botcol[2]+1;col<=topcol[3];col+=1)x[Q*boundaryrow[col]+col]
=((1.0*col-1.0*topcol[3])/(1.0*topcol[3]-1.0*botcol[2]))*perpdist[2];
for(col=topcol[3]+1;col<=botcol[3];col+=1)x[Q*boundaryrow[col]+col]
=((1.0*col-1.0*topcol[3])/(1.0*botcol[3]-1.0*topcol[3]))*perpdist[0];

for(col=0;col<=numcols;col+=1)
{
x[Q*numrows+col]=a*cos(2.0*PI*(col-topcol[0])/(1.0*numcols));
y[Q*numrows+col]=a*sin(2.0*PI*(col-topcol[0])/(1.0*numcols));
}

dothebound();

for(col=0;col<=numcols;col+=1)
{
for(row=boundaryrow[col]+1;row<=numrows-1;row+=1)
{
x[Q*row+col]=(x[Q*boundaryrow[col]+col]*(1.0*numrows-1.0*row)
              +x[Q*numrows+col]*(1.0*row-1.0*boundaryrow[col]))
              /(1.0*numrows-1.0*boundaryrow[col]);
y[Q*row+col]=(y[Q*boundaryrow[col]+col]*(1.0*numrows-1.0*row)
              +y[Q*numrows+col]*(1.0*row-1.0*boundaryrow[col]))
              /(1.0*numrows-1.0*boundaryrow[col]);
}
}
}

void dothebound(void)
{
for(k=0;k<=3;k+=1)
{
if(k==0){for(col=0;col<=topcol[0]-1;col+=1)setbound();}
else {for(col=botcol[k-1];col<=topcol[k]-1;col+=1)setbound();}

for(col=topcol[k];col<=botcol[k];col+=1)setbound();
}

for(row=0;row<=botrow[3];row+=1)
{
x[Q*row+numcols]=perpdist[0];
y[Q*row]=-perpdist[3];
}
}

void setbound(void)
{
for(row=0;row<=boundaryrow[col];row+=1)
{
if(k==0){x[Q*row+col]=
perpdist[0];if(row!=boundaryrow[col])y[Q*row+col]=y[Q*boundaryrow[col]
+col];}
```

```
if(k==1){y[Q*row+col]=
perpdist[1];if(row!=boundaryrow[col])x[Q*row+col]=x[Q*boundaryrow[col]
+col];}
if(k==2){x[Q*row+col]=-
perpdist[2];if(row!=boundaryrow[col])y[Q*row+col]=y[Q*boundaryrow[col]
+col];}
if(k==3){y[Q*row+col]=-
perpdist[3];if(row!=boundaryrow[col])x[Q*row+col]=x[Q*boundaryrow[col]
+col];}
}
}

void typicalnodexy(void)
{
if(col>=1)whichcol=col-1;else whichcol=numcols-1;

tx=x[Q*row+col];
ty=y[Q*row+col];
polar();

lambda=1.0-0.0004*(1.5*numrows-1.0*row)*(1.0+0.05*(1.0-
cos(2.0*theta)));;
mu=2.0-lambda;

tx=(x[Q*row+col+1]+x[Q*row+whichcol])/4.0-x[Q*row+col]/2.0;
ty=(y[Q*row+col+1]+y[Q*row+whichcol])/4.0-y[Q*row+col]/2.0;
tz=(z[Q*row+col+1]+z[Q*row+whichcol])/4.0-z[Q*row+col]/2.0;

tx+=(lambda*x[Q*(row+1)+col]+mu*x[Q*(row-1)+col])/4.0-
x[Q*row+col]/2.0;
ty+=(lambda*y[Q*(row+1)+col]+mu*y[Q*(row-1)+col])/4.0-
y[Q*row+col]/2.0;
tz+=(lambda*z[Q*(row+1)+col]+mu*z[Q*(row-1)+col])/4.0-
z[Q*row+col]/2.0;

xmovement[Q*row+col]=overrelax*tx;
ymovement[Q*row+col]=overrelax*ty;
zmovement[Q*row+col]=overrelax*tz;

tempfloat=xmovement[Q*row+col]*xnormal[Q*row+col]
          +ymovement[Q*row+col]*ynormal[Q*row+col]
          +zmovement[Q*row+col]*znormal[Q*row+col];

xmovement[Q*row+col]-=xnormal[Q*row+col]*tempfloat;
ymovement[Q*row+col]-=ynormal[Q*row+col]*tempfloat;
zmovement[Q*row+col]-=znormal[Q*row+col]*tempfloat;
}

void calczanalytic(void)
{
for(row=0;row<=numrows;row+=1)
{
for(col=0;col<=numcols;col+=1)
{
for(wherex=0;wherex<=2;wherex+=1)
{
for(wherey=0;wherey<=2;wherey+=1)
{
if(wherex==1||wherey==1)
{
if(wherex==1&&wherey==1&&row<=
boundaryrow[col])storezvalue[wherex][wherey]=edgeheight;
else
{
if(wherex==1&&wherey==1&&row==
```

```
numrows)storezvalue[wherex][wherey]=centreheight;
else
{
xvalue=x[Q*row+col]+(1.0+1.0*wherex)*delta;
yvalue=y[Q*row+col]+(1.0+1.0*wherey)*delta;
findtheheight();
storezvalue[wherex][wherey]=zvalue;
}
}
}
}
}
}
z[Q*row+col]=storezvalue[1][1];
dzbydx[Q*row+col]=(storezvalue[2][1]-storezvalue[0][1])/(2.0*delta);
dzbydy[Q*row+col]=(storezvalue[1][2]-storezvalue[1][0])/(2.0*delta);
}
}
}

for(row=numrows;row>=0;row-=1)
{
for(col=0;col<=numcols;col+=1)
{
if((col==0||col==botcol[0]||col==botcol[1]||col==botcol[2]||col==botco
l[3])
&&row<=boundaryrow[col])
{
dzbydx[Q*row+col]=dzbydx[Q*(row+1)+col];
dzbydy[Q*row+col]=dzbydy[Q*(row+1)+col];
}
if(((col>botcol[0]&&col<botcol[1])||(col>botcol[2]&&col<botcol[3]))
&&row<=boundaryrow[col])dzbydx[Q*row+col]=0.0;
if(((col>botcol[1]&&col<botcol[2])||(col>0&&col<botcol[0]))
&&row<=boundaryrow[col])dzbydy[Q*row+col]=0.0;
}
}
}

void calcxy(void)
{
cyclesSoFar=0;
previousKE=0.0;
for(col=0;col<=numcols-1;col+=1)
{
for(row=boundaryrow[col]+1;row<=numrows-1;row+=1)
{
prevxmovement[Q*row+col]=0.0;
prevymovement[Q*row+col]=0.0;
prevzmovement[Q*row+col]=0.0;
}
}
}

findxycyclesnum:
printf("Number of xy cycles? ");
scanf("%d",&maxnumxycycles);
if(maxnumxycycles==0)goto xyover;
printf("Intermediate cycles? ");
scanf("%d",&intermediate);
printf("Over relaxation factor? ");
scanf("%lf",&overrelax);
printf("Carry over factor? ");
scanf("%lf",&carryover);

intermediatecycle=0;
for(cycle=1;cycle<=maxnumxycycles;cycle+=1)
{
```

```
cyclesSoFar+=1;
if(intermediatecycle==0)findnormals();
intermediatecycle+=1;
if(intermediatecycle>intermediate)intermediatecycle=0;

KE=0.0;

for(col=0;col<=numcols-1;col+=1)
{
for(row=boundaryrow[col]+1;row<=numrows-1;row+=1)
{
typicalnodexy();
xmovement[Q*row+col]+=carryover*prevxmovement[Q*row+col];
ymovement[Q*row+col]+=carryover*prevymovement[Q*row+col];
zmovement[Q*row+col]+=carryover*prevzmovement[Q*row+col];

KE+=xmovement[Q*row+col]*xmovement[Q*row+col]
     +ymovement[Q*row+col]*ymovement[Q*row+col]
     +zmovement[Q*row+col]*zmovement[Q*row+col];

x[Q*row+col]+=xmovement[Q*row+col];
y[Q*row+col]+=ymovement[Q*row+col];
z[Q*row+col]+=zmovement[Q*row+col];

if(col==topcol[0]||col==topcol[2])y[Q*row+col]=0.0;
if(col==topcol[1]||col==topcol[3])x[Q*row+col]=0.0;

prevxmovement[Q*row+col]=xmovement[Q*row+col];
prevymovement[Q*row+col]=ymovement[Q*row+col];
prevzmovement[Q*row+col]=zmovement[Q*row+col];
}
}

//printf("Cycle   %d   KE   %le   Increase   %le\n",cyclesSoFar,KE,KE-
previousKE);

if(KE<previousKE||cycle==1)
printf("Cycle %d KE %le\n",cyclesSoFar,KE);

if(KE<previousKE)
{
previousKE=0.0;
for(col=0;col<=numcols-1;col+=1)
{
for(row=boundaryrow[col]+1;row<=numrows-1;row+=1)
{
prevxmovement[Q*row+col]=0.0;
prevymovement[Q*row+col]=0.0;
prevzmovement[Q*row+col]=0.0;
}
}
}
else previousKE=KE;

for(row=0;row<=numrows;row+=1)
{
x[Q*row+numcols]=x[Q*row];
y[Q*row+numcols]=y[Q*row];
z[Q*row+numcols]=z[Q*row];
}
}

goto findxycyclesnum;
xyover:;
}
```

```
void polar(void)
{
r=sqrt(tx*tx+ty*ty);
if(fabs(ty)>fabs(tx)){theta=acos(tx/r);if(ty<0.0)theta=-theta;}
else{theta=asin(ty/r);if(tx<0.0)theta=PI-theta;}
}

void findtheheight(void)
{
tx=xvalue;
ty=yvalue;
polar();

tempfloat=(1.0-xvalue/perpdist[0])*(1.0+xvalue/perpdist[2])
         *(1.0-yvalue/perpdist[1])*(1.0+yvalue/perpdist[3]);

zvalue=(centreheight-edgeheight)*tempfloat/
((1.0-(a/r)*(xvalue/perpdist[0]))*(1.0+(a/r)*(xvalue/perpdist[2]))
*(1.0-(a/r)*(yvalue/perpdist[1]))*(1.0+(a/r)*(yvalue/perpdist[3])))
+edgeheight;

grad=0.0;

xdistance=perpdist[0]-xvalue;ydistance=perpdist[1]-yvalue;
grad+=sqrt(xdistance*xdistance+ydistance*ydistance)/(xdistance*ydistan
ce);

xdistance=perpdist[0]-xvalue;ydistance=perpdist[3]+yvalue;
grad+=sqrt(xdistance*xdistance+ydistance*ydistance)/(xdistance*ydistan
ce);

xdistance=perpdist[2]+xvalue;ydistance=perpdist[3]+yvalue;
grad+=sqrt(xdistance*xdistance+ydistance*ydistance)/(xdistance*ydistan
ce);

xdistance=perpdist[2]+xvalue;ydistance=perpdist[1]-yvalue;
grad+=sqrt(xdistance*xdistance+ydistance*ydistance)/(xdistance*ydistan
ce);

ConeAmount=0.5;                                  //This controls "foldiness"
at corners
zvalue+=ConeAmount*
(
3.5*(1.0+cos(2.0*theta))/2.0
+3.0*(1.0-cos(2.0*theta))/2.0
+0.3*sin(theta)
)
*(1.0-(a/r))/grad;

zvalue+=(1.0-ConeAmount)*
(
(35.0+10.0*tempfloat)*(1.0+cos(2.0*theta))/2.0
//=1 when theta=0,180;=0 when theta=90,270
+24.0*(((1.0-cos(2.0*theta))/2.0)+sin(theta))/2.0
//=0 when theta=0,180, 270;=1 when theta=90
+(7.5+12.0*tempfloat)*(((1.0-cos(2.0*theta))/2.0)-sin(theta))/2.0
//=0 when theta=0,180,  90;=1 when theta=270
-1.6//Lowers whole thing
)
*(r/a-1.0)*tempfloat;

//This controls short span:
zvalue-=10.0*((1.0+cos(2.0*theta))/2.0)*(r/a-1.0)*tempfloat;
```

```
//This raises roof over North Portico:
zvalue+=10.0*pow(((((1.0-cos(2.0*theta))/2.0)+sin(theta))/2.0),2.0)
*(r/a-1.0)*tempfloat*(1.0-3.0*(r/a-1.0)*tempfloat);

//This raises roof over South Portico:
zvalue+=2.5*pow(((((1.0-cos(2.0*theta))/2.0)-sin(theta))/2.0),2.0)
*pow((r/a-1.0),3.0)*tempfloat;

//This controls corners:
//zvalue-=5.0*((1.0-cos(4.0*theta))/2.0)*(r/a-1.0)*tempfloat;

//This is Waagner Buro's four corners raise
Waagner=14.0;
zvalue+=1.05*((1.0-(a/r))/grad)*
(exp(Waagner*(-(1.0-xvalue/perpdist[0])))+
 exp(Waagner*(-(1.0+xvalue/perpdist[2]))))*
(exp(Waagner*(-(1.0-yvalue/perpdist[1])))+
 exp(Waagner*(-(1.0+yvalue/perpdist[3]))));
}

void findnormals(void)
{
if(normalcalc==1)
{
for(col=0;col<=numcols-1;col+=1)
{
for(row=0;row<=numrows;row+=1)
{
if(row<=boundaryrow[col])z[Q*row+col]=edgeheight;
else
{
if(row==numrows)z[Q*row+col]=centreheight;
else
{
xvalue=x[Q*row+col];
yvalue=y[Q*row+col];
findtheheight();
z[Q*row+col]=zvalue;
}
}
}
}
for(col=0;col<=numcols-1;col+=1)
{
if(col>=1)whichcol=col-1;else whichcol=numcols-1;
for(row=boundaryrow[col]+1;row<=numrows-1;row+=1)
{
xnormal[Q*row+col]=(y[Q*(row+1)+col]-y[Q*(row-
1)+col])*(z[Q*row+col+1]-z[Q*row+whichcol])-(z[Q*(row+1)+col]-
z[Q*(row-1)+col])*(y[Q*row+col+1]-y[Q*row+whichcol]);
ynormal[Q*row+col]=(z[Q*(row+1)+col]-z[Q*(row-
1)+col])*(x[Q*row+col+1]-x[Q*row+whichcol])-(x[Q*(row+1)+col]-
x[Q*(row-1)+col])*(z[Q*row+col+1]-z[Q*row+whichcol]);
znormal[Q*row+col]=(x[Q*(row+1)+col]-x[Q*(row-
1)+col])*(y[Q*row+col+1]-y[Q*row+whichcol])-(y[Q*(row+1)+col]-
y[Q*(row-1)+col])*(x[Q*row+col+1]-x[Q*row+whichcol]);

tempfloat=xnormal[Q*row+col]*xnormal[Q*row+col]
                +ynormal[Q*row+col]*ynormal[Q*row+col]
                +znormal[Q*row+col]*znormal[Q*row+col];

tempfloat=sqrt(tempfloat);

xnormal[Q*row+col]=xnormal[Q*row+col]/tempfloat;
ynormal[Q*row+col]=ynormal[Q*row+col]/tempfloat;
```

```
znormal[Q*row+col]=znormal[Q*row+col]/tempfloat;
}
}
}
}

else
{
for(col=0;col<=numcols-1;col+=1)
{
for(row=boundaryrow[col]+1;row<=numrows-1;row+=1)
{
xnormal[Q*row+col]=0.0;
ynormal[Q*row+col]=0.0;
znormal[Q*row+col]=1.0;
}
}
}
}
```

D3.2 British Museum Roof computer program to generate spiral

```
#include <math.h>
#include <stdio.h>
#define   surfacenum     40000
#define   linenum        20000
#define   maxnumcolsp1     801
#define   step               2
#define   dxf                1
#define    extrabits          1//0 for no extra nodes and members, eg
columns, snow gallery etc.
#define   dxfmembershape     0
//#define   CableControl       2//0 for no cables, 1 for corner ties

extern void SetUpDrawingArea();
extern void FinishOffDrawing();

extern void dxfSetUp();
extern void dxf3DGridFace();
extern void dxf3DRadFace();
extern void dxf3DClockFace();
extern void dxf3DAntiFace();
extern void dxf3DBoundFace();
extern void dxfCentreLine();
extern void dxfSystemLine();
extern void dxfNormal();
extern void dxfNodeNo();
extern void dxfNodeDepths();
extern void dxfMemberNo();
extern void dxfFinishOff();

void grid(void);
void WriteCoordorLoadData(void);
void writecoordsorloads(void);
void findnormals(void);
void ExtraMemberDimensions(void);
void Draw(void);
void glass(void);
void makeglass(void);
void makemembers(void);
void MakeaMemFace(void);

float
PI,x[surfacenum],y[surfacenum],z[surfacenum],ytoplot[surfacenum],
      dzbydx[surfacenum],dzbydy[surfacenum],
```

```
      zload,liveloadfactor,deadloadfactor,
      normalx[surfacenum],normaly[surfacenum],normalz[surfacenum],
      glassarea[surfacenum],
      scale,
      tx,ty,tz,xyz[3],
      a,perpdist[4],r,psi,
      tempfloat,minmemberlength,

      xstart,ystart,zstart,xfinis,yfinis,zfinis,
      ax,ay,az,bx,by,bz,px,py,pz,qx,qy,qz,
      textsize,
      boundmemdepth[50],boundmemwidth[50],

      averagememberdepth[300],averagememberwidth[300],
      topbotT[300],sideT[300],
      crosssectA[300],majorI[300],minorI[300],J[300],


betapointx[linenum],betapointy[linenum],betapointz[linenum],betaangle[
2][linenum],
      lengthmem[linenum],
      radius,theta,cornerdistance,
      sign1,sign2,sign3,sign4,


glassx1,glassy1,glassz1,glassx2,glassy2,glassz2,glassx3,glassy3,glassz
3,
      asq,bsq,csq,area,

      secondtempfloat,polartheta,polarphi[surfacenum],
      memlensq,perpx[2],perpy[2],perpz[2],zprojx,zprojy,zprojz,
      glassweightperunitarea,snowload,
      totalglassarea,
      rectthingy,

      tpx,tpy,tpz,xx,xy,xz,yx,yy,yz,
      nodedepthfromfile[surfacenum],normaldrawinglength,

xcentreline[surfacenum],ycentreline[surfacenum],zcentreline[surfacenum
],

      cornertrussx,cornertrussy,cornertrussz;

int   row,col,numrows,numcols,intx,inty,
      topcol[4],botcol[4],toprow[4],botrow[4],minbotrow,maxtoprow,
      boundaryrow[maxnumcolsp1],
      diagonal,minnumrows,
      rowstodraw,colstodraw,colstodrawperside[4],noddy,Q,
      k,
      whichcol,nextcol,
      row1,col1,row2,col2,row3,col3,
      trow,tcol,
      nodeno,memberno,//tempmembertype,
      totalnodeno,totalmemberno,
      tempint,
      NumberOfMainRoofMembers,NumberOfMainRoofNodes,StillOnMainRoof,
      glassstart1,glassstart2,PaneNo,totalpaneno,
      coordsorloads,cornertrussint;
      //laststeelmember;

short int whichnode[surfacenum],
          end1[linenum],end2[linenum],
          tempshort,
          origend1[linenum],origend2[linenum],
          MemberDirect[linenum],
```

```
          noderow[linenum],nodecol[linenum];

RGBColor    Colour;

FILE  *DataFile,*OutPutFile,
      *OtherDataFile,*OtherOutPutFile,
      *MemberProps,*AreaFile,
      *nodedepths,*nodmid,*SlopeFile,
      *Connect;

void main(void)
{
//startagain:
StillOnMainRoof=1;

glassweightperunitarea=0.55;
snowload=0.6;
liveloadfactor=3.0;
deadloadfactor=3.0;
totalglassarea=0.0;
PI=4.0*atan(1.0);
textsize=0.1;

DataFile=fopen("DimData","r");
fscanf(DataFile,"%f",&tempfloat);a=tempfloat;
for(k=0;k<=3;k+=1){fscanf(DataFile,"%f",&tempfloat);perpdist[k]=tempfl
oat;}
fclose(DataFile);

cornerdistance=sqrt((perpdist[0]+perpdist[2])*(perpdist[0]+perpdist[2]
)/4.0+perpdist[3]*perpdist[3]);

DataFile=fopen("Coordinates","r");
fscanf(DataFile,"%d%d",&numrows,&numcols);
Q=numcols+1;
for(k=0;k<=3;k+=1)fscanf(DataFile,"%d",&topcol[k]);
for(k=0;k<=3;k+=1)fscanf(DataFile,"%d",&botcol[k]);
for(row=0;row<=numrows;row+=1)
{
for(col=0;col<=numcols;col+=1)
{
fscanf(DataFile,"%d%d",&trow,&tcol);
fscanf(DataFile,"%d%e%e%e%e%e",
  &boundaryrow[tcol],
  &x[Q*trow+tcol],
  &y[Q*trow+tcol],
  &z[Q*trow+tcol],
  &dzbydx[Q*trow+tcol],
  &dzbydy[Q*trow+tcol]);
}
}
fclose(DataFile);

if(dxf==1)dxfSetUp();

findnormals();
ExtraMemberDimensions();

for(col=0;col<=numcols-1;col+=1)
{
for(row=0;row<=numrows-1;row+=1)
{
if(row>=boundaryrow[col]||row>=boundaryrow[col+1])
{
if(dxf==1)dxf3DGridFace(
```

```
x[Q*(row+0)+(col+0)],y[Q*(row+0)+(col+0)],z[Q*(row+0)+(col+0)],
x[Q*(row+0)+(col+1)],y[Q*(row+0)+(col+1)],z[Q*(row+0)+(col+1)],
x[Q*(row+1)+(col+1)],y[Q*(row+1)+(col+1)],z[Q*(row+1)+(col+1)],
x[Q*(row+1)+(col+0)],y[Q*(row+1)+(col+0)],z[Q*(row+1)+(col+0)]);
}
}
}

nodedepths=fopen("NodeDepths","r");
OutPutFile=fopen("SystemPoints","w");
SlopeFile=fopen("NodalSlopes","w");
coordsorloads=0;
WriteCoordorLoadData();
totalnodeno=nodeno;
fclose(OutPutFile);
fclose(SlopeFile);
fclose(nodmid);
fclose(nodedepths);

grid();//contains writing connectivity

OutPutFile=fopen("TriangleNodes","w");
glass();
fclose(OutPutFile);

/*for(tempint=4;tempint<=9;tempint+=1)
{
for(memberno=1;memberno<=totalmemberno;memberno+=1)
{
if(MemberDirect[memberno]==tempint)
{
membertype[memberno]=tempmembertype;
if(CableControl==0)
if(MemberDirect[memberno]==6)laststeelmember=memberno;
//THIS APPLIES WITHOUT CORNER BRACE
if(CableControl==1)
if(MemberDirect[memberno]==7)laststeelmember=memberno;
//THIS APPLIES WITH CORNER BRACE
}
}
tempmembertype+=1;
}*/

OutPutFile=fopen("Members","w");
Connect=fopen("Connectivity","w");
OtherOutPutFile=fopen("OtherMembers","w");
for(memberno=1;memberno<=totalmemberno;memberno+=1)
{
makemembers();

tx=(x[origend1[memberno]]+x[origend2[memberno]])/2.0;
ty=(y[origend1[memberno]]+y[origend2[memberno]])/2.0;
if(dxf==1&&memberno<=NumberOfMainRoofMembers)
dxfMemberNo(memberno,tx,ty,textsize);
if(dxf==1&&memberno<=NumberOfMainRoofMembers)
dxfSystemLine(x[origend1[memberno]],y[origend1[memberno]],z[origend1[m
emberno]],

x[origend2[memberno]],y[origend2[memberno]],z[origend2[memberno]]);

tempfloat=(betaangle[0][memberno]+betaangle[1][memberno])/2.0;
if(memberno<=NumberOfMainRoofMembers)
{
fprintf(OutPutFile,"%5d%8hd%8hd%8hd%12.3f\r",memberno,
whichnode[origend1[memberno]],whichnode[origend2[memberno]],
```

```
MemberDirect[memberno],tempfloat);
fprintf(Connect,"%5d%8hd%8hd\r",memberno,
whichnode[origend1[memberno]],whichnode[origend2[memberno]]);
}
else
fprintf(OtherOutPutFile,"%5d%8hd%8hd%8hd%12.3f\r",memberno,
whichnode[origend1[memberno]],whichnode[origend2[memberno]],
MemberDirect[memberno],tempfloat);
}

for(col=0;col<=numcols-1;col+=1)
{
nextcol=col+1;if(nextcol==numcols)nextcol=0;
dxfSystemLine(x[Q*boundaryrow[col]+col],y[Q*boundaryrow[col]+col],
z[Q*boundaryrow[col]+col],
x[Q*boundaryrow[nextcol]+nextcol],y[Q*boundaryrow[nextcol]+nextcol],
z[Q*boundaryrow[nextcol]+nextcol]);
}

for(col=0;col<=numcols-1;col+=1)
{
nextcol=col+1;if(nextcol==numcols)nextcol=0;
dxfSystemLine(x[Q*numrows+col],y[Q*numrows+col],z[Q*numrows+col],

x[Q*numrows+nextcol],y[Q*numrows+nextcol],z[Q*numrows+nextcol]);
}

fclose(OutPutFile);
fclose(OtherOutPutFile);
fclose(Connect);

//membertype[totalmemberno+1]=0;

AreaFile=fopen("AreaData","w");
fprintf(AreaFile,"  Node Surface   Slope\r");
fprintf(AreaFile,"        Area\r");
coordsorloads=1;WriteCoordorLoadData();
fclose(AreaFile);

if(dxf==1)dxfFinishOff();
OutPutFile=fopen("Numbers","w");
fprintf(OutPutFile,"%d    %d    %d\r",

NumberOfMainRoofNodes,NumberOfMainRoofMembers,totalpaneno);
fclose(OutPutFile);

OutPutFile=fopen("MirrorNodes","w");
for(nodeno=1;nodeno<=NumberOfMainRoofNodes;nodeno+=1)
{
row=noderow[nodeno];
col=nodecol[nodeno];
col=botcol[2]-col;
if(col<0)col+=numcols;
fprintf(OutPutFile,"%d  %hd\r",nodeno,whichnode[Q*row+col]);
}

fclose(OutPutFile);

Draw();
printf("\nFinished\n");
}

void grid(void)
{
memberno=0;
```

```
for(col=0;col<=numcols-step;col+=step)
{
if(
col!=topcol[0]&&
col!=topcol[1]&&
col!=topcol[2]&&
col!=topcol[3]&&
col!=topcol[1]-2*step&&col!=topcol[1]+2*step&&
col!=topcol[3]-2*step&&col!=topcol[3]+2*step
)
{
for(row=boundaryrow[col]+2*step;row<=numrows;row+=2*step)
{
origend1[memberno+1]=Q*(row-2*step)+col;
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=1;memberno+=1;
}
}
else
{
origend1[memberno+1]=Q*boundaryrow[col]+col;
origend2[memberno+1]=Q*(boundaryrow[col]+step)+col;
MemberDirect[memberno+1]=1;memberno+=1;
for(row=boundaryrow[col]+3*step;row<=numrows;row+=2*step)
{
origend1[memberno+1]=Q*(row-2*step)+col;
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=1;memberno+=1;
}
}
}

diagonal=step;
for(col=0;col<=numcols-step;col+=step)
{
if(diagonal==step)diagonal=0;else diagonal=step;
for(row=diagonal;row<=numrows-step;row+=2*step)
{
if(row>=boundaryrow[col]&&row+step!=boundaryrow[col+step])
{
origend1[memberno+1]=Q*row+col;
if(col!=numcols-step)origend2[memberno+1]=Q*(row+step)+col+step;
else origend2[memberno+1]=Q*(row+step);
MemberDirect[memberno+1]=2;
memberno+=1;
}
}
}

diagonal=0;
for(col=step;col<=numcols;col+=step)
{
if(diagonal==step)diagonal=0;else diagonal=step;
for(row=diagonal;row<=numrows-step;row+=2*step)
{
if(row>=boundaryrow[col]&&row+step!=boundaryrow[col-step])
{
if(col!=numcols)origend1[memberno+1]=Q*row+col;
else origend1[memberno+1]=Q*row;
origend2[memberno+1]=Q*(row+step)+col-step;
MemberDirect[memberno+1]=3;
memberno+=1;
}
}
}
```

```
NumberOfMainRoofMembers=memberno;

if(extrabits!=0)
{
//Rectanglar in plan edge beam i.e. outside edge beam
col=0;
row=numrows+1;
origend1[memberno+1]=Q*row+col;
for(col=step;col<=numcols;col+=step)
{
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=4;memberno+=1;
origend1[memberno+1]=origend2[memberno];
}

//Perimeter edge beam stubs
for(col=0;col<=numcols-step;col+=step)
{
origend1[memberno+1]=Q*boundaryrow[col]+col;;
origend2[memberno+1]=Q*(numrows+1)+col;
MemberDirect[memberno+1]=5;memberno+=1;
}

//Reading Room edge beam
row=numrows+2;
col=0;
origend1[memberno+1]=Q*row+col;
for(col=step;col<=numcols;col+=step)
{
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=6;memberno+=1;
origend1[memberno+1]=origend2[memberno];
}

//Reading Room edge beam stubs
for(col=step;col<=numcols-step;col+=2*step)
{
  row=numrows;origend1[memberno+1]=Q*row+col;
row=numrows+2;origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=7;memberno+=1;
}

//Reading Room columns
for(col=2*step;col<=numcols-4*step;col+=6*step)
{
row=numrows+2;origend1[memberno+1]=Q*row+col;
row=numrows+3;origend2[memberno+1]=Q*row+col;
printf("Column is member number %d\n",memberno+1);
MemberDirect[memberno+1]=8;memberno+=1;
}

/*//Corner Braces
if(CableControl==1)
{
for(k=0;k<=3;k+=1)
{
col=botcol[k]-2*step;
row=boundaryrow[col];
origend1[memberno+1]=Q*row+col;
col+=4*step;if(col>=numcols)col-=numcols;
row=boundaryrow[col];
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=7;memberno+=1;
}
```

```
}*/

//This is for corner trusses
for(k=0;k<=3;k+=1)
{
for(cornertrussint=-2;cornertrussint<=2;cornertrussint+=1)
{
col=botcol[k]+cornertrussint*step;
if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
row=numrows+1;
origend1[memberno+1]=Q*row+col;
row=numrows+4;
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=9;memberno+=1;
}

for(cornertrussint=-2;cornertrussint<=-1;cornertrussint+=1)
{
col=botcol[k]+(cornertrussint+1)*step;
if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
row=numrows+1;
origend1[memberno+1]=Q*row+col;
col=botcol[k]+cornertrussint*step;
if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
row=numrows+4;
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=9;memberno+=1;
}

for(cornertrussint=1;cornertrussint<=2;cornertrussint+=1)
{
col=botcol[k]+(cornertrussint-1)*step;
if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
row=numrows+1;
origend1[memberno+1]=Q*row+col;
col=botcol[k]+cornertrussint*step;
if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
row=numrows+4;
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=9;memberno+=1;
}

for(cornertrussint=-1;cornertrussint<=2;cornertrussint+=1)
{
col=botcol[k]+(cornertrussint-1)*step;
if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
row=numrows+4;
origend1[memberno+1]=Q*row+col;
col=botcol[k]+cornertrussint*step;
if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
row=numrows+4;
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=9;memberno+=1;
}

for(cornertrussint=-2;cornertrussint<=2;cornertrussint+=1)
{
col=botcol[k]+cornertrussint*step;
if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
row=boundaryrow[col]+2*step;
origend1[memberno+1]=Q*row+col;
row=numrows+4;
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=10;memberno+=1;
}
```

```
}

//Concrete snow Gallery
row=numrows+3;
col=numcols-4*step;
origend1[memberno+1]=Q*row+col;
for(col=2*step;col<=numcols-4*step;col+=6*step)
{
origend2[memberno+1]=Q*row+col;
MemberDirect[memberno+1]=11;memberno+=1;
origend1[memberno+1]=origend2[memberno];
}
}

totalmemberno=memberno;
}

void WriteCoordorLoadData(void)
{
nodeno=0;

for(col=0;col<=numcols-step;col+=step)
{
if(
col!=topcol[0]&&
col!=topcol[1]&&
col!=topcol[2]&&
col!=topcol[3]&&
col!=topcol[1]-2*step&&col!=topcol[1]+2*step&&
col!=topcol[3]-2*step&&col!=topcol[3]+2*step
)
{
for(row=boundaryrow[col];row<=numrows;row+=2*step)writecoordsorloads()
;
}
else
{
row=boundaryrow[col];
writecoordsorloads();
for(row=boundaryrow[col]+step;row<=numrows;row+=2*step)writecoordsorlo
ads();
}
}

if(coordsorloads==0)
{
NumberOfMainRoofNodes=nodeno;
StillOnMainRoof=0;
}

if(extrabits!=0&&coordsorloads==0)
{
row=numrows+1;//This is to give nodes at centreline of perimeter edge
beam
for(col=0;col<=numcols-step;col+=step)
{
x[Q*row+col]=x[Q*boundaryrow[col]+col];
y[Q*row+col]=y[Q*boundaryrow[col]+col];
z[Q*row+col]=z[Q*boundaryrow[col]+col]-0.175;          //Modified 27-1-
99
                if(col<=botcol[0])x[Q*row+col]-=0.075;//Modified  27-1-
99
if(col>=botcol[0]&&col<=botcol[1])y[Q*row+col]-=0.075;//Modified  27-1-
99
```

```
if(col>=botcol[1]&&col<=botcol[2])x[Q*row+col]+=0.075;//Modified  27-1-
99
        if(col==0||col>=botcol[2])y[Q*row+col]+=0.075;//Modified  27-1-
99
writecoordsorloads();
}

row=numrows+2;//This is to give nodes at centreline of Reading Room
edge beam
for(col=0;col<=numcols-step;col+=step)
{
x[Q*row+col]=x[Q*(row-2)+col];
y[Q*row+col]=y[Q*(row-2)+col];
z[Q*row+col]=z[Q*(row-2)+col]-0.175;
x[Q*row+col]+=0.175*x[Q*row+col]/a;//Modified 27-1-99
y[Q*row+col]+=0.175*y[Q*row+col]/a;//Modified 27-1-99
writecoordsorloads();
}

row=numrows+3;//Note this is for extra nodes below ring
for(col=2*step;col<=numcols-4*step;col+=6*step)
{
x[Q*row+col]=x[Q*(row-1)+col];
y[Q*row+col]=y[Q*(row-1)+col];
z[Q*row+col]=z[Q*(row-1)+col]-1.15;
writecoordsorloads();
}

row=numrows+4;//Note this is for corner trusses
for(k=0;k<=3;k+=1)
{
col=botcol[k];if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
if(k==0||k==3)cornertrussx=x[Q*(numrows+1)+col]+1.5;
else          cornertrussx=x[Q*(numrows+1)+col]-1.5;
if(k==0||k==1)cornertrussy=y[Q*(numrows+1)+col]+1.5;
else          cornertrussy=y[Q*(numrows+1)+col]-1.5;
cornertrussz=z[Q*(numrows+1)+col];

for(cornertrussint=-2;cornertrussint<=-1;cornertrussint+=1)
{
col=botcol[k]+cornertrussint*step;if(col<0)col+=numcols;
if(col>=numcols)col-=numcols;
if(k==0){x[Q*row+col]=cornertrussx+0.0*cornertrussint;
y[Q*row+col]=cornertrussy+3.0*cornertrussint;}
if(k==1){x[Q*row+col]=cornertrussx-3.0*cornertrussint;
y[Q*row+col]=cornertrussy+0.0*cornertrussint;}
if(k==2){x[Q*row+col]=cornertrussx+0.0*cornertrussint;
y[Q*row+col]=cornertrussy-3.0*cornertrussint;}
if(k==3){x[Q*row+col]=cornertrussx+3.0*cornertrussint;
y[Q*row+col]=cornertrussy+0.0*cornertrussint;}
z[Q*row+col]=cornertrussz;
writecoordsorloads();
}

col=botcol[k];if(col<0)col+=numcols;if(col>=numcols)col-=numcols;
x[Q*row+col]=cornertrussx;
y[Q*row+col]=cornertrussy;
z[Q*row+col]=cornertrussz;
writecoordsorloads();

for(cornertrussint=1;cornertrussint<=2;cornertrussint+=1)
{
col=botcol[k]+cornertrussint*step;if(col<0)col+=numcols;
if(col>=numcols)col-=numcols;
```

```
if(k==0){x[Q*row+col]=cornertrussx-
3.0*cornertrussint;y[Q*row+col]=cornertrussy+0.0*cornertrussint;}
if(k==1){x[Q*row+col]=cornertrussx+0.0*cornertrussint;
y[Q*row+col]=cornertrussy-3.0*cornertrussint;}
if(k==2){x[Q*row+col]=cornertrussx+3.0*cornertrussint;
y[Q*row+col]=cornertrussy+0.0*cornertrussint;}
if(k==3){x[Q*row+col]=cornertrussx+0.0*cornertrussint;
y[Q*row+col]=cornertrussy+3.0*cornertrussint;}
z[Q*row+col]=cornertrussz;
writecoordsorloads();
}
}

}
}

void Draw(void)
{
scale=3.5;

DataFile=fopen("NodeCentres","r");
for(tempint=1;tempint<=totalnodeno;tempint+=1)
{
fscanf(DataFile,"%d",&nodeno);
fscanf(DataFile,"%e%e%e",&x[nodeno],&y[nodeno],&z[nodeno]);
}
fclose(DataFile);

DataFile=fopen("Members","r");
OtherDataFile=fopen("OtherMembers","r");
for(tempint=1;tempint<=totalmemberno;tempint+=1)
{
if(tempint<=NumberOfMainRoofMembers)
{
fscanf(DataFile,"%d",&memberno);
fscanf(DataFile,"%hd%hd%hd%f",&end1[memberno],&end2[memberno],
&MemberDirect[memberno],&tempfloat);
}
else
{
fscanf(OtherDataFile,"%d",&memberno);
fscanf(OtherDataFile,"%hd%hd%hd%f",&end1[memberno],&end2[memberno],
&MemberDirect[memberno],&tempfloat);
}

}
fclose(DataFile);
fclose(OtherDataFile);

for(memberno=1;memberno<=totalmemberno;memberno+=1)
{
tempfloat=sqrt(
(x[end1[memberno]]-x[end2[memberno]])*(x[end1[memberno]]-
x[end2[memberno]])+
(y[end1[memberno]]-y[end2[memberno]])*(y[end1[memberno]]-
y[end2[memberno]])+
(z[end1[memberno]]-z[end2[memberno]])*(z[end1[memberno]]-
z[end2[memberno]]));
if(memberno==1)minmemberlength=tempfloat;
else
{
if(tempfloat<minmemberlength)minmemberlength=tempfloat;
}
}
printf("Minimum member length = %fmm\n",1000.0*minmemberlength);
```

```
SetUpDrawingArea(317,218);
for(memberno=1;memberno<=totalmemberno;memberno+=1)
{
Colour.red=65535.0*0.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0.
0;
if(MemberDirect[memberno]==1)
{Colour.red=65535.0*1.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0
.0;}
if(MemberDirect[memberno]==2)
{Colour.red=65535.0*0.0;Colour.green=65535.0*1.0;Colour.blue=65535.0*0
.0;}
if(MemberDirect[memberno]==3)
{Colour.red=65535.0*0.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*1
.0;}
if(MemberDirect[memberno]==4)
{Colour.red=65535.0*0.0;Colour.green=65535.0*1.0;Colour.blue=65535.0*1
.0;}
if(MemberDirect[memberno]==5)
{Colour.red=65535.0*1.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*1
.0;}
if(MemberDirect[memberno]==6)
{Colour.red=65535.0*1.0;Colour.green=65535.0*1.0;Colour.blue=65535.0*0
.0;}
if(MemberDirect[memberno]==7)
{Colour.red=65535.0*0.5;Colour.green=65535.0*0.0;Colour.blue=65535.0*0
.0;}
if(MemberDirect[memberno]==8)
{Colour.red=65535.0*0.0;Colour.green=65535.0*0.5;Colour.blue=65535.0*0
.0;}
if(MemberDirect[memberno]==9)
{Colour.red=65535.0*0.0;Colour.green=65535.0*0.0;Colour.blue=65535.0*0
.5;}
if(MemberDirect[memberno]==10)
{Colour.red=65535.0*0.0;Colour.green=65535.0*0.5;Colour.blue=65535.0*0
.5;}
if(MemberDirect[memberno]==11)
{Colour.red=65535.0*0.5;Colour.green=65535.0*0.0;Colour.blue=65535.0*0
.5;}
if(MemberDirect[memberno]==2)
{Colour.red=65535.0*0.5;Colour.green=65535.0*0.5;Colour.blue=65535.0*0
.0;}

RGBForeColor(&Colour);

intx=317+scale*x[end1[memberno]];inty=218-scale*y[end1[memberno]];
MoveTo(intx,inty);
intx=317+scale*x[end2[memberno]];inty=218-scale*y[end2[memberno]];
LineTo(intx,inty);
}
FinishOffDrawing();
}

void findnormals(void)
{
for(col=0;col<=numcols;col+=1)
{
for(row=0;row<=numrows;row+=1)
{
normalx[Q*row+col]=-dzbydx[Q*row+col];
normaly[Q*row+col]=-dzbydy[Q*row+col];
normalz[Q*row+col]=1.0;

tempfloat=sqrt(
normalx[Q*row+col]*normalx[Q*row+col]+
```

```
normaly[Q*row+col]*normaly[Q*row+col]+
normalz[Q*row+col]*normalz[Q*row+col]);

normalx[Q*row+col]=normalx[Q*row+col]/tempfloat;
normaly[Q*row+col]=normaly[Q*row+col]/tempfloat;
normalz[Q*row+col]=normalz[Q*row+col]/tempfloat;
}
}
}

void glass(void)
{
PaneNo=0;
glassstart1=0;
glassstart2=step;
for(col=0;col<=numcols-step;col+=step)
{
if(glassstart1==0)glassstart1=step;else glassstart1=0;
if(glassstart2==0)glassstart2=step;else glassstart2=0;
for(row=glassstart1;row<=numrows;row+=2*step)
{
row1=row;row2=row+step;row3=row-step;
col1=col+step;col2=col;col3=col;
if(glassstart1==step&&row2>numrows){row2=numrows;col2=col-
step;if(col2<0)col2+=numcols;}
if(row3==boundaryrow[col3]-step)row3=boundaryrow[col3];
if(row1>=boundaryrow[col+step]&&row3>=boundaryrow[col]&&
row2<=numrows)makeglass();
}
for(row=glassstart2;row<=numrows;row+=2*step)
{
row1=row;row2=row-step;row3=row+step;
col1=col;col2=col+step;col3=col+step;
if(glassstart2==0&&row3>numrows){row3=numrows;col3=col+2*step;}
if(row2==boundaryrow[col2]-step)row2=boundaryrow[col2];
if(row1>=boundaryrow[col]&&row2>=boundaryrow[col+step]&&
row3<=numrows)makeglass();
}
totalpaneno=PaneNo;
}
}

void writecoordsorloads(void)
{
nodeno+=1;

if(coordsorloads==0)
{
if(nodeno==1)nodmid=fopen("NodeCentres","w");

tempfloat=sqrt(dzbydx[Q*row+col]*dzbydx[Q*row+col]+
dzbydy[Q*row+col]*dzbydy[Q*row+col]);
if(tempfloat>1.0e-6)
{
if(fabs(dzbydx[Q*row+col])<fabs(dzbydy[Q*row+col]))
{
polartheta=acos(-dzbydx[Q*row+col]/tempfloat);
if((-dzbydy[Q*row+col])<0)polartheta=-polartheta;
}
else
{
polartheta=asin(-dzbydy[Q*row+col]/tempfloat);
if((-dzbydx[Q*row+col])<0)polartheta=PI-polartheta;
if(polartheta>PI)polartheta-=2.0*PI;
}
```

```
}
else polartheta=0.0;
polarphi[Q*row+col]=atan(tempfloat);
polartheta=(180.0/PI)*polartheta;
polarphi[Q*row+col]=(180.0/PI)*polarphi[Q*row+col];

if(StillOnMainRoof==1)
{
noderow[nodeno]=row;
nodecol[nodeno]=col;
fprintf(OutPutFile,"%4d%12.4f%12.4f%12.4f%12.4f%12.4f\r",
nodeno,x[Q*row+col],y[Q*row+col],z[Q*row+col],
polartheta,polarphi[Q*row+col]);
fprintf(SlopeFile,"%d    %e    %e\n",nodeno,dzbydx[Q*row+col],
dzbydy[Q*row+col]);
}

normaldrawinglength=1.0;
if(dxf==1&&nodeno<=NumberOfMainRoofNodes)dxfNormal(x[Q*row+col],
y[Q*row+col],z[Q*row+col],
x[Q*row+col]+normaldrawinglength*sin(polarphi[Q*row+col]*PI/180.0)*
cos(polartheta*PI/180.0),
y[Q*row+col]+normaldrawinglength*sin(polarphi[Q*row+col]*PI/180.0)*
sin(polartheta*PI/180.0),
z[Q*row+col]+normaldrawinglength*cos(polarphi[Q*row+col]*PI/180.0));

if(StillOnMainRoof==1)
{
fscanf(nodedepths,"%d",&tempint);
if(tempint!=nodeno)printf("Problem with node numbers in depth
file\n");
fscanf(nodedepths,"%f",&nodedepthfromfile[nodeno]);
nodedepthfromfile[nodeno]=nodedepthfromfile[nodeno]/1000.0;
//printf("%d  %f\n",nodeno,nodedepthfromfile[nodeno]);
xcentreline[Q*row+col]=x[Q*row+col]-
(nodedepthfromfile[nodeno]/2.0)*sin(polarphi[Q*row+col]*PI/180.0)*
cos(polartheta*PI/180.0);
ycentreline[Q*row+col]=y[Q*row+col]-
(nodedepthfromfile[nodeno]/2.0)*sin(polarphi[Q*row+col]*PI/180.0)*
sin(polartheta*PI/180.0);
zcentreline[Q*row+col]=z[Q*row+col]-
(nodedepthfromfile[nodeno]/2.0)*cos(polarphi[Q*row+col]*PI/180.0);
}

else
{
xcentreline[Q*row+col]=x[Q*row+col];
ycentreline[Q*row+col]=y[Q*row+col];
zcentreline[Q*row+col]=z[Q*row+col];
}

if(col==0)
{
xcentreline[Q*row+numcols]=xcentreline[Q*row+col];
ycentreline[Q*row+numcols]=ycentreline[Q*row+col];
zcentreline[Q*row+numcols]=zcentreline[Q*row+col];
}

fprintf(nodmid,"%4d%12.4f%12.4f%12.4f\r",nodeno,xcentreline[Q*row+col]
,
ycentreline[Q*row+col],zcentreline[Q*row+col]);

tx=x[Q*row+col];ty=y[Q*row+col];
if(dxf==1&&StillOnMainRoof==1)dxfNodeNo(nodeno,tx,ty,textsize);
```

```
if(dxf==1&&StillOnMainRoof==1)dxfNodeDepths(nodedepthfromfile[nodeno],
tx,ty);
whichnode[Q*row+col]=nodeno;
if(col==0)whichnode[Q*row+numcols]=nodeno;

glassarea[whichnode[Q*row+col]]=0.0;
if(col==0)glassarea[whichnode[Q*row+numcols]]=0.0;
}
else
{
radius=sqrt(x[Q*row+col]*x[Q*row+col]+y[Q*row+col]*y[Q*row+col]);
theta=acos(x[Q*row+col]/radius);if(y[Q*row+col]<0.0)theta=-theta;

if(glassarea[whichnode[Q*row+col]]!=0.0)
fprintf(AreaFile,"%5d%8.3f%8.1f\r",nodeno,glassarea[whichnode[Q*row+co
l]],
polarphi[Q*row+col]);

zload=-glassweightperunitarea*glassarea[whichnode[Q*row+col]];
totalglassarea+=glassarea[whichnode[Q*row+col]];

if(polarphi[Q*row+col]<=30.0)
zload-=snowload*glassarea[whichnode[Q*row+col]];
if(30.0<polarphi[Q*row+col]&&polarphi[Q*row+col]<=60.0)
zload-=snowload*glassarea[whichnode[Q*row+col]]*
(1.0-(polarphi[Q*row+col]-30.0)/30.0);
if(zload!=0.0)
{
zload=zload/liveloadfactor;
}
}
}

void makeglass(void)
{
PaneNo+=1;

fprintf(OutPutFile,"%5d%8hd%8hd%8hd%8hd\r",
PaneNo,whichnode[Q*row1+col1],whichnode[Q*row2+col2],
whichnode[Q*row3+col3]);

glassx1=x[Q*row1+col1];
glassy1=y[Q*row1+col1];
glassz1=z[Q*row1+col1];
glassx2=x[Q*row2+col2];
glassy2=y[Q*row2+col2];
glassz2=z[Q*row2+col2];
glassx3=x[Q*row3+col3];
glassy3=y[Q*row3+col3];
glassz3=z[Q*row3+col3];

asq=(glassx2-glassx3)*(glassx2-glassx3)+(glassy2-glassy3)*(glassy2-
glassy3)+
    (glassz2-glassz3)*(glassz2-glassz3);
bsq=(glassx3-glassx1)*(glassx3-glassx1)+(glassy3-glassy1)*(glassy3-
glassy1)+
    (glassz3-glassz1)*(glassz3-glassz1);
csq=(glassx1-glassx2)*(glassx1-glassx2)+(glassy1-glassy2)*(glassy1-
glassy2)+
    (glassz1-glassz2)*(glassz1-glassz2);
                    area=asq*bsq+bsq*csq+csq*asq;
                    area=2.0*area-(asq*asq+bsq*bsq+csq*csq);
if(area<0.0)printf("Error! Negative Area!\n");
area=sqrt(area)/4.0;
glassarea[whichnode[Q*row1+col1]]+=area/3.0;
```

```
glassarea[whichnode[Q*row2+col2]]+=area/3.0;
glassarea[whichnode[Q*row3+col3]]+=area/3.0;
}

void ExtraMemberDimensions(void)
{
for(tempint=0;tempint<=20;tempint+=1)
{boundmemwidth[tempint]=0.0;boundmemdepth[tempint]=0.0;}

//Note boundmemwidth and boundmemdepth are only used for drawing.
//If the value is 0.0, they are not drawn.
//Perimeter NOTE Beta has to be increased by 90 degrees:
//boundmemwidth[1]=0.15;boundmemdepth[1]=0.3;
//Inner ring:
boundmemwidth[3]=0.35; boundmemdepth[3]=0.35;
//Columns:
//boundmemwidth[3]=0.0; boundmemdepth[3]=0.0;
/*//Corner ties:
if(CableControl==1)
{
tempfloat=sqrt(1450e-6*170.0/210.0);
boundmemwidth[4]=tempfloat;boundmemdepth[4]=tempfloat;
}*/
//Concrete ring:
//boundmemwidth[6]=0.0; boundmemdepth[6]=0.0;
}

void makemembers(void)
{
xstart=xcentreline[origend1[memberno]];
ystart=ycentreline[origend1[memberno]];
zstart=zcentreline[origend1[memberno]];
xfinis=xcentreline[origend2[memberno]];
yfinis=ycentreline[origend2[memberno]];
zfinis=zcentreline[origend2[memberno]];

betapointx[memberno]=(xstart+xfinis)/2.0+10.0*
(normalx[origend1[memberno]]+normalx[origend2[memberno]])/2.0;
betapointy[memberno]=(ystart+yfinis)/2.0+10.0*
(normaly[origend1[memberno]]+normaly[origend2[memberno]])/2.0;
betapointz[memberno]=(zstart+zfinis)/2.0+10.0*
(normalz[origend1[memberno]]+normalz[origend2[memberno]])/2.0;

memlensq=(xfinis-xstart)*(xfinis-xstart)
        +(yfinis-ystart)*(yfinis-ystart)
        +(zfinis-zstart)*(zfinis-zstart);

lengthmem[memberno]=sqrt(memlensq);

tempfloat=((xfinis-xstart)*normalx[origend1[memberno]]+
           (yfinis-ystart)*normaly[origend1[memberno]]+
           (zfinis-zstart)*normalz[origend1[memberno]])/memlensq;
perpx[0]=normalx[origend1[memberno]]-tempfloat*(xfinis-xstart);
perpy[0]=normaly[origend1[memberno]]-tempfloat*(yfinis-ystart);
perpz[0]=normalz[origend1[memberno]]-tempfloat*(zfinis-zstart);

tempfloat=((xfinis-xstart)*normalx[origend2[memberno]]+
           (yfinis-ystart)*normaly[origend2[memberno]]+
           (zfinis-zstart)*normalz[origend2[memberno]])/memlensq;
perpx[1]=normalx[origend2[memberno]]-tempfloat*(xfinis-xstart);
perpy[1]=normaly[origend2[memberno]]-tempfloat*(yfinis-ystart);
perpz[1]=normalz[origend2[memberno]]-tempfloat*(zfinis-zstart);

tempfloat=(zfinis-zstart)/memlensq;
zprojx=-tempfloat*(xfinis-xstart);
```

```
zprojy=-tempfloat*(yfinis-ystart);
zprojz=1.0-tempfloat*(zfinis-zstart);

for(tempint=0;tempint<=1;tempint+=1)
{
if(MemberDirect[memberno]<=3)
{
tempfloat=sqrt(memlensq*
(perpx[tempint]*perpx[tempint]+
 perpy[tempint]*perpy[tempint]+
 perpz[tempint]*perpz[tempint])*
(zprojx*zprojx+zproiy*zproiy+zprojz*zprojz));
if(tempfloat>1.0e-2)
{
betaangle[tempint][memberno]=((zprojy*perpz[tempint]-
zprojz*perpy[tempint])*
(xfinis-xstart)+(zprojz*perpx[tempint]-zprojx*perpz[tempint])*
(yfinis-ystart)+(zprojx*perpy[tempint]-zproiy*perpx[tempint])*
(zfinis-zstart))/tempfloat;
if(betaangle[tempint][memberno]>1.0)
{
printf("Beta        angle        problem        %f                        %d
%d\n",betaangle[tempint][memberno],tempint,memberno);
betaangle[tempint][memberno]=1.0;
}
if(betaangle[tempint][memberno]<-1.0)
{
printf("Beta        angle        problem        %f                        %d
%d\n",betaangle[tempint][memberno],tempint,memberno);
betaangle[tempint][memberno]=-1.0;
}
betaangle[tempint][memberno]=(180.0/PI)*asin(betaangle[tempint][member
no]);
}
}
else betaangle[tempint][memberno]=0.0;//This for boundary and other
members
}

if(MemberDirect[memberno]<=3)
{
ax=(yfinis-ystart)*normalz[origend1[memberno]]-(zfinis-zstart)*
normaly[origend1[memberno]];
ay=(zfinis-zstart)*normalx[origend1[memberno]]-(xfinis-xstart)*
normalz[origend1[memberno]];
az=(xfinis-xstart)*normaly[origend1[memberno]]-(yfinis-ystart)*
normalx[origend1[memberno]];

tempfloat=0.08/(2.0*sqrt(ax*ax+ay*ay+az*az));

ax=ax*tempfloat;
ay=ay*tempfloat;
az=az*tempfloat;

tempfloat=nodedepthfromfile[whichnode[origend1[memberno]]];
//printf("%f\n",tempfloat);

bx=normalx[origend1[memberno]]*tempfloat;
by=normaly[origend1[memberno]]*tempfloat;
bz=normalz[origend1[memberno]]*tempfloat;
}
else
{
ax=(yfinis-ystart)*1.0-(zfinis-zstart)*0.0;
ay=(zfinis-zstart)*0.0-(xfinis-xstart)*1.0;
```

```
az=(xfinis-xstart)*0.0-(yfinis-ystart)*0.0;

tempfloat=boundmemwidth[MemberDirect[memberno]-3]/(2.0*
sqrt(ax*ax+ay*ay+az*az));

ax=ax*tempfloat;
ay=ay*tempfloat;
az=az*tempfloat;

tempfloat=boundmemdepth[MemberDirect[memberno]-3];

bx=0.0*tempfloat;
by=0.0*tempfloat;
bz=1.0*tempfloat;
}

if(MemberDirect[memberno]<=3)
{
px=(yfinis-ystart)*normalz[origend2[memberno]]-(zfinis-zstart)*
normaly[origend2[memberno]];
py=(zfinis-zstart)*normalx[origend2[memberno]]-(xfinis-xstart)*
normalz[origend2[memberno]];
pz=(xfinis-xstart)*normaly[origend2[memberno]]-(yfinis-ystart)*
normalx[origend2[memberno]];

tempfloat=0.08/(2.0*sqrt(px*px+py*py+pz*pz));

px=px*tempfloat;
py=py*tempfloat;
pz=pz*tempfloat;

tempfloat=nodedepthfromfile[whichnode[origend2[memberno]]];
//printf("%f\n",tempfloat);

qx=normalx[origend2[memberno]]*tempfloat;
qy=normaly[origend2[memberno]]*tempfloat;
qz=normalz[origend2[memberno]]*tempfloat;
}
else
{
px=(yfinis-ystart)*1.0-(zfinis-zstart)*0.0;
py=(zfinis-zstart)*0.0-(xfinis-xstart)*1.0;
pz=(xfinis-xstart)*0.0-(yfinis-ystart)*0.0;

tempfloat=boundmemwidth[MemberDirect[memberno]-3]/(2.0*
sqrt(px*px+py*py+pz*pz));

px=px*tempfloat;
py=py*tempfloat;
pz=pz*tempfloat;

tempfloat=boundmemdepth[MemberDirect[memberno]-3];

qx=0.0*tempfloat;
qy=0.0*tempfloat;
qz=1.0*tempfloat;
}

if(MemberDirect[memberno]<=3||boundmemdepth[MemberDirect[memberno]-
3]!=0.0)
{
sign1=-1.0;sign2=+0.0;sign3=+1.0;sign4=+0.0;MakeaMemFace();
sign1=+1.0;sign2=+0.0;sign3=+1.0;sign4=-1.0;MakeaMemFace();
sign1=+1.0;sign2=-1.0;sign3=-1.0;sign4=-1.0;MakeaMemFace();
sign1=-1.0;sign2=+0.0;sign3=-1.0;sign4=-1.0;MakeaMemFace();
```

```
}

if(dxf==1)dxfCentreLine(MemberDirect[memberno],MemberDirect[memberno],
xstart,ystart,zstart,xfinis,yfinis,zfinis);
}

void MakeaMemFace(void)
{
if(dxfmembershape==1)
{
if(MemberDirect[memberno]==1)
{
if(dxf==1)dxf3DRadFace(
xstart+sign1*ax+sign2*bx,ystart+sign1*ay+sign2*by,zstart+sign1*az+sign
2*bz,
xfinis+sign1*px+sign2*qx,yfinis+sign1*py+sign2*qy,zfinis+sign1*pz+sign
2*qz,
xfinis+sign3*px+sign4*qx,yfinis+sign3*py+sign4*qy,zfinis+sign3*pz+sign
4*qz,
xstart+sign3*ax+sign4*bx,ystart+sign3*ay+sign4*by,zstart+sign3*az+sign
4*bz);
}

if(MemberDirect[memberno]==2)
{
if(dxf==1)dxf3DClockFace(
xstart+sign1*ax+sign2*bx,ystart+sign1*ay+sign2*by,zstart+sign1*az+sign
2*bz,
xfinis+sign1*px+sign2*qx,yfinis+sign1*py+sign2*qy,zfinis+sign1*pz+sign
2*qz,
xfinis+sign3*px+sign4*qx,yfinis+sign3*py+sign4*qy,zfinis+sign3*pz+sign
4*qz,
xstart+sign3*ax+sign4*bx,ystart+sign3*ay+sign4*by,zstart+sign3*az+sign
4*bz);
}

if(MemberDirect[memberno]==3)
{
if(dxf==1)dxf3DAntiFace(
xstart+sign1*ax+sign2*bx,ystart+sign1*ay+sign2*by,zstart+sign1*az+sign
2*bz,
xfinis+sign1*px+sign2*qx,yfinis+sign1*py+sign2*qy,zfinis+sign1*pz+sign
2*qz,
xfinis+sign3*px+sign4*qx,yfinis+sign3*py+sign4*qy,zfinis+sign3*pz+sign
4*qz,
xstart+sign3*ax+sign4*bx,ystart+sign3*ay+sign4*by,zstart+sign3*az+sign
4*bz);
}

if(MemberDirect[memberno]>=4)
{
if(dxf==1)dxf3DBoundFace(
xstart+sign1*ax+sign2*bx,ystart+sign1*ay+sign2*by,zstart+sign1*az+sign
2*bz,
xfinis+sign1*px+sign2*qx,yfinis+sign1*py+sign2*qy,zfinis+sign1*pz+sign
2*qz,
xfinis+sign3*px+sign4*qx,yfinis+sign3*py+sign4*qy,zfinis+sign3*pz+sign
4*qz,
xstart+sign3*ax+sign4*bx,ystart+sign3*ay+sign4*by,zstart+sign3*az+sign
4*bz);
}
}
}
```